

Refine Search

Search Results -

Terms	Documents
L55 not @py>1998	. 5

Database:

US Pre-Grant Publication Full-Text Database
 US Patents Full-Text Database
 US OCR Full-Text Database
 EPO Abstracts Database
 JPO Abstracts Database
 Derwent World Patents Index
 IBM Technical Disclosure Bulletins

Search:

Search History

DATE: Wednesday, March 07, 2007 [Purge Queries](#) [Printable Copy](#) [Create Case](#)

<u>Set</u> <u>Name</u>	<u>Query</u>	<u>Hit</u> <u>Count</u>	<u>Set</u> <u>Name</u> result set
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L56</u>	L55 not @py>1998	5	<u>L56</u>
<u>L55</u>	L54 and electronic near document	361	<u>L55</u>
<u>L54</u>	L53 and ("uniform resource locator" or "url")	2860	<u>L54</u>
<u>L53</u>	version and time adj1 stamp\$2	14502	<u>L53</u>
	<i>DB=USPT; PLUR=YES; OP=OR</i>		
<u>L52</u>	("6125371")[URPN]	8	<u>L52</u>
<u>L51</u>	(5745905 5317731 5956713)![PN]	3	<u>L51</u>
<u>L50</u>	("6125371")[PN]	1	<u>L50</u>
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L49</u>	6125371.pn.	2	<u>L49</u>
	<i>DB=USPT; PLUR=YES; OP=OR</i>		
<u>L48</u>	("5991802")[URPN]	43	<u>L48</u>
	(5761683 5745754 5751956 5907847 5608907 5745681 5745764		

<u>L47</u>	5761421 5706507 5726979)! [PN]	10	<u>L47</u>
<u>L46</u>	("5991802") [PN]	1	<u>L46</u>
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L45</u>	5991802.pn.	2	<u>L45</u>
	<i>DB=USPT; PLUR=YES; OP=OR</i>		
<u>L44</u>	("6006227") [URPN]	43	<u>L44</u>
<u>L43</u>	(5402526 5649182 5530859 5063495 5140676 5890177 5835129 5448729 5535063 5625818 5159669)! [PN]	11	<u>L43</u>
<u>L42</u>	("6006227") [PN]	1	<u>L42</u>
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L41</u>	6006227.pn.	2	<u>L41</u>
	<i>DB=USPT; PLUR=YES; OP=OR</i>		
<u>L40</u>	("5991773") [URPN]	7	<u>L40</u>
<u>L39</u>	(5778231 5678042 5659735 5519865 4907188 5802299)! [PN]	6	<u>L39</u>
<u>L38</u>	("5991773") [PN]	1	<u>L38</u>
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L37</u>	5991773.pn.	2	<u>L37</u>
	<i>DB=EPAB,JPAB; PLUR=YES; OP=OR</i>		
<u>L36</u>	l29 and ("uniform resource locator" or "url")	0	<u>L36</u>
<u>L35</u>	l30 and ("uniform resource locator" or "url")	0	<u>L35</u>
<u>L34</u>	l24 and L29	0	<u>L34</u>
<u>L33</u>	l24 and L30	0	<u>L33</u>
<u>L32</u>	l19 and L29	0	<u>L32</u>
<u>L31</u>	l19 and L30	0	<u>L31</u>
<u>L30</u>	l13 and 707.clas.	81	<u>L30</u>
<u>L29</u>	l14 and 715.clas.	658	<u>L29</u>
	<i>DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR</i>		
<u>L28</u>	l26 not @py>1998	1	<u>L28</u>
<u>L27</u>	l25 not @py>1998	3	<u>L27</u>
<u>L26</u>	L24 and 707.clas.	143	<u>L26</u>
<u>L25</u>	L24 and 715.clas.	69	<u>L25</u>
<u>L24</u>	L23 and (electronic near document or electronic with document or electronic adj document)	580	<u>L24</u>
<u>L23</u>	L22 and version	2860	<u>L23</u>
<u>L22</u>	L21 and ("uniform resource locator" or "url")	4400	<u>L22</u>
<u>L21</u>	(time adj1 stamp\$2)	38862	<u>L21</u>
<u>L20</u>	L19 not @py>1998	1	<u>L20</u>
<u>L19</u>	L18 and ("uniform resource locator" or "url")	206	<u>L19</u>
<u>L18</u>	L17 and (domain near name or domain with name or domain adj name)	256	<u>L18</u>
<u>L17</u>	l16 and (timestamp or time-stamp or time with stamp or time adj stamp)	1585	<u>L17</u>
<u>L16</u>	L15 and version	10491	<u>L16</u>
	(electronic near document or electronic with document or electronic adj		

L15 document)
L14 715.clas.
L13 707.clas.
L12 707/501.1
L11 707/500.1
L10 707/1
L9 707/513
L8 707/511
L7 707/203
L6 715/203
L5 715/500.1
L4 715/501.1
L3 715/500
L2 715/513
L1 715/511

38970 L15
28787 L14
41558 L13
761 L12
447 L11
9209 L10
2868 L9
554 L8
3905 L7
2 L6
1219 L5
1641 L4
1375 L3
3099 L2
459 L1

END OF SEARCH HISTORY


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

composing url searches with timestamps

SEARCH

THE ACM DIGITAL LIBRARY


[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

 Terms used **composing url searches with timestamps**

 Found **19,462** of **198,310**

Sort results by

relevance


[Save results to a Binder](#)
[Try an Advanced Search](#)
[Try this search in The ACM Guide](#)

Display results

expanded form


[Search Tips](#)
☐ Open results in a new window

Results 1 - 20 of 200

 Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale ☐ ☐ ☐ ☐ ☐

1 [Measuring and characterizing end-to-end Internet service performance](#)



Ludmila Cherkasova, Yun Fu, Wenting Tang, Amin Vahdat

 November 2003 **ACM Transactions on Internet Technology (TOIT)**, Volume 3 Issue 4

Publisher: ACM Press

Full text available: pdf(1.46 MB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Fundamental to the design of reliable, high-performance network services is an understanding of the performance characteristics of the service as perceived by the client population as a whole. Understanding and measuring such end-to-end service performance is a challenging task. Current techniques include periodic sampling of service characteristics from strategic locations in the network and instrumenting Web pages with code that reports client-perceived latency back to a performance server. Li ...

Keywords: End-to-end service performance, QoS, network packet traces, passive monitoring, reconstruction of web page composition, web site performance

2 [Shallow NLP techniques for internet search](#)



Alex Penev, Raymond Wong

 January 2006 **Proceedings of the 29th Australasian Computer Science Conference - Volume 48 ACSC '06**

Publisher: Australian Computer Society, Inc.

 Full text available: pdf(422.28 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Information Retrieval (IR) is a major component in many of our daily activities, with perhaps its most prominent role manifested in search engines. Today's most advanced engines use the keyword-based ("bag of words") paradigm, which concedes some inherent disadvantages. We believe that natural language (NL) is a more user-oriented, context-preservative and intuitive mechanism for web search. In this paper, we explore shallow NLP techniques to support a range of NL queries over an existing keyword ...

Keywords: google, information retrieval, natural language, processing

3 [Fast detection of communication patterns in distributed executions](#)



Thomas Kunz, Michiel F. H. Seuren


 November 1997 **Proceedings of the 1997 conference of the Centre for Advanced**

Studies on Collaborative research CASCON '97**Publisher:** IBM PressFull text available:  [pdf\(4.21 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

4 Extracting usability information from user interface events

David M. Hilbert, David F. Redmiles


December 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 4**Publisher:** ACM PressFull text available:  [pdf\(1.50 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Modern window-based user interface systems generate user interface events as natural products of their normal operation. Because such events can be automatically captured and because they indicate user behavior with respect to an application's user interface, they have long been regarded as a potentially fruitful source of information regarding application usage and usability. However, because user interface events are typically voluminous and rich in detail, automated support is generally ...

Keywords: human-computer interaction, sequential data analysis, usability testing, user interface event monitoring

5 Evaluating implicit measures to improve web search

Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, Thomas White


April 2005 **ACM Transactions on Information Systems (TOIS)**, Volume 23 Issue 2**Publisher:** ACM PressFull text available:  [pdf\(2.52 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Of growing interest in the area of improving the search experience is the collection of implicit user behavior measures (implicit measures) as indications of user interest and user satisfaction. Rather than having to submit explicit user feedback, which can be costly in time and resources and alter the pattern of use within the search experience, some research has explored the collection of implicit measures as an efficient and useful alternative to collecting explicit measure of interest from u ...

Keywords: Implicit measures, explicit feedback, explicit ratings, prediction model, search sessions, user interest, user satisfaction

6 Verifying security protocols as planning in logic programming

Luigia Carlucci Aiello, Fabio Massacci

October 2001 **ACM Transactions on Computational Logic (TOCL)**, Volume 2 Issue 4**Publisher:** ACM PressFull text available:  [pdf\(305.94 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

We illustrate *ALSP* (Action Language for Security Protocol), a declarative executable specification language for planning attacks to security protocols. *ALSP* is based on logic programming with negation as failure, and with stable model semantics. In *ALSP* we can

give a declarative specification of a protocol with the natural semantics of send and receive actions which can be performed in parallel. By viewing a protocol trace as a plan to a ...

Keywords: AI planning, logic programming, security protocols, specification language

7 Affinity-based management of main memory database clusters



Minwen Ji

November 2002 **ACM Transactions on Internet Technology (TOIT)**, Volume 2 Issue 4

Publisher: ACM Press

Full text available: pdf(553.96 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We study management strategies for main memory database clusters that are interposed between Internet applications and back-end databases as content caches. The task of management is to allocate data across individual cache databases and to route queries to the appropriate databases for execution. The goal is to maximize effective cache capacity and to minimize synchronization cost. We propose an affinity-based management system for main memory database clusters (*ALBUM*). *ALBUM* executes ea ...

Keywords: Main memory database, clustering, database administration, database cluster, file organization, query affinity, scalability

8 Papers: Event-based multimedia chronicling systems



Pilho Kim, Ullas Gargi, Ramesh Jain

November 2005 **Proceedings of the 2nd ACM workshop on Continuous archival and retrieval of personal experiences CARPE '05**

Publisher: ACM Press

Full text available: pdf(791.39 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This paper presents a new framework for multimedia electronic chronicling systems. Its approach uses events as its driving force for heterogeneous information processing. Specifically, this new approach first separates symbols and data, then puts events between them to make a distinct connection. In addition, this approach provides spatio-temporal-semantic relations networks to map high-level semantic user queries into low-level queries that a machine can compute. The innovative user interfaces are ...

Keywords: action capture, browse, chronicle, context modeling, event, information, information sharing, interaction, interfaces, mobile, monitoring, presentation systems, publication, publishing, retrieval, search, tag

9 Characterizing a national community web



Daniel Gomes, Mário J. Silva

August 2005 **ACM Transactions on Internet Technology (TOIT)**, Volume 5 Issue 3

Publisher: ACM Press

Full text available: pdf(364.77 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article presents a characterization of the community Web of the people of Portugal. We defined criteria for delimiting this Web based on our past experience of crawling pages related to Portugal and collected over 3.2 million documents from 46,000 sites satisfying those criteria. Our characterization was derived from this crawl. We describe the rules that we established for defining the boundaries of this community Web and the methodology used to gather statistics. Statistics cover the numb ...

Keywords: Portuguese Web, Web characterization, Web communities, Web measurements

10 Boosting the performance of Web search engines: Caching and prefetching query results by exploiting historical usage data



Tiziano Fagni, Raffaele Perego, Fabrizio Silvestri, Salvatore Orlando

January 2006 **ACM Transactions on Information Systems (TOIS)**, Volume 24 Issue 1

Publisher: ACM Press

Full text available: pdf(668.69 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article discusses efficiency and effectiveness issues in caching the results of queries submitted to a Web search engine (WSE). We propose SDC (Static Dynamic Cache), a new caching strategy aimed to efficiently exploit the temporal and spatial locality present in the stream of processed queries. SDC extracts from historical usage data the results of the most frequently submitted queries and stores them in a *static, read-only* portion of the cache. The remaining entries of the c ...

Keywords: Caching, Web search engines, multithreading

11 An architecture for secure wide-area service discovery

Todd D. Hodes, Steven E. Czerwinski, Ben Y. Zhao, Anthony D. Joseph, Randy H. Katz

March 2002 **Wireless Networks**, Volume 8 Issue 2/3

Publisher: Kluwer Academic Publishers

Full text available: pdf(365.68 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The widespread deployment of inexpensive communications technology, computational resources in the networking infrastructure, and network-enabled end devices poses an interesting problem for end users: how to locate a particular network service or device out of hundreds of thousands of accessible services and devices. This paper presents the architecture and implementation of a secure wide-area Service Discovery Service (SDS). Service providers use the SDS to advertise descriptions of available ...

Keywords: location services, name lookup, network protocols, service discovery

12 A survey of Web metrics



Devanshu Dhyani, Wee Keong Ng, Sourav S. Bhowmick

December 2002 **ACM Computing Surveys (CSUR)**, Volume 34 Issue 4

Publisher: ACM Press

Full text available: pdf(289.28 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The unabated growth and increasing significance of the World Wide Web has resulted in a flurry of research activity to improve its capacity for serving information more effectively. But at the heart of these efforts lie implicit assumptions about "quality" and "usefulness" of Web resources and services. This observation points towards measurements and models that quantify various attributes of web sites. The science of measuring all aspects of information, especially its storage and retrieval or ...

Keywords: Information theoretic, PageRank, Web graph, Web metrics, Web page similarity, quality metrics

13 Web and design: Generating custom notification histories by tracking visual differences between web page visits

Saul Greenberg, Michael Boyle

June 2006 **Proceedings of the 2006 conference on Graphics interface GI '06**

Publisher: Canadian Information Processing Society

Full text available:  pdf(586.37 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

We contribute a method that lets people create a visual history of custom notifications to track personally meaningful changes to web pages. Notifications are assembled as a collage of regions extracted from the fully rendered (bitmap) representation of the web pages. They are triggered when visual changes between successive visits are detected within regions. To use the system, a person specifies a notification by clipping personally interesting regions from the bitmap representation of a web p ...

Keywords: information customization, notifications

14 Best paper session: best paper candidates: Modelling information persistence on the web



Daniel Gomes, Mário J. Silva

July 2006 **Proceedings of the 6th international conference on Web engineering ICWE '06**

Publisher: ACM Press

Full text available:  pdf(187.75 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Models of web data persistency are essential tools for the design of efficient information extraction systems that repeatedly collect and process the data. This study models the persistence of web data through the measurement of URL and content persistence across several snapshots of a national community web, collected for 3 years. We found that the lifetimes of URLs and contents are modelled by logarithmic functions. We gathered statistics on the structure of the web, identified reasons for URL death ...

Keywords: content persistence, tomba, url persistence

15 Attacks and cryptanalysis: Puppetnets: misusing web browsers as a distributed attack infrastructure



V. T. Lam, S. Antonatos, P. Akritidis, K. G. Anagnostakis

October 2006 **Proceedings of the 13th ACM conference on Computer and communications security CCS '06**

Publisher: ACM Press

Full text available:  pdf(871.35 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Most of the recent work on Web security focuses on preventing attacks that *directly* harm the browser's host machine and user. In this paper we attempt to quantify the threat of browsers being *indirectly* misused for attacking third parties. Specifically, we look at how the existing Web infrastructure (e.g., the languages, protocols, and security policies) can be exploited by malicious Web sites to remotely instruct browsers to orchestrate actions including denial of service attacks, ...

Keywords: distributed attacks, malicious software, web security

16 Multimedia and visualization: Dynamic structuring of web information for access visualization




Jess Y. S. Mak, Hong Va Leong, Alvin T. S. Chan

March 2002 **Proceedings of the 2002 ACM symposium on Applied computing SAC '02****Publisher:** ACM PressFull text available:  [pdf\(765.23 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The Internet has led to the formation of a global information infrastructure. To explore a web site, a site map would be useful as a short cut for a user to locate for the target information in a structured and efficient manner, rather than drilling into the web site following hyperlinks, reading possibly irrelevant information. Useless information impacts a mobile web environment, where mobile clients are only connected with unreliable wireless channels of limited bandwidth. Structured web page ...

Keywords: DOM, VRML, XML, visualization, web document structure17 Reviewed articles: Measuring the evolution of transport protocols in the internet

Alberto Medina, Mark Allman, Sally Floyd

April 2005 **ACM SIGCOMM Computer Communication Review**, Volume 35 Issue 2**Publisher:** ACM PressFull text available:  [pdf\(1.48 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In this paper we explore the evolution of both the Internet's most heavily used transport protocol, TCP, and the current network environment with respect to how the network's evolution ultimately impacts end-to-end protocols. The traditional end-to-end assumptions about the Internet are increasingly challenged by the introduction of intermediary network elements (middleboxes) that intentionally or unintentionally prevent or alter the behavior of end-to-end communications. This paper provides mea ...

Keywords: Internet, TCP, evolution, middleboxes18 An architecture for a secure service discovery service


Steven E. Czerwinski, Ben Y. Zhao, Todd D. Hodes, Anthony D. Joseph, Randy H. Katz

August 1999 **Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking MobiCom '99****Publisher:** ACM PressFull text available:  [pdf\(1.47 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)19 Cluster-based scalable network services

Armando Fox, Steven D. Gribble, Yatin Chawathe, Eric A. Brewer, Paul Gauthier

October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles SOSP '97**, Volume 31 Issue 5**Publisher:** ACM PressFull text available:  [pdf\(2.42 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)20 Web-conscious storage management for web proxies

Evangelos P. Markatos, Dionisios N. Pnevmatikatos, Michail D. Flouris, Manolis G. H. Katevenis

December 2002 **IEEE/ACM Transactions on Networking (TON)**, Volume 10 Issue 6**Publisher:** IEEE PressFull text available:  [pdf\(603.11 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Many proxy servers are limited by their file I/O needs. Even when a proxy is configured with sufficient I/O hardware, the file system software often fails to provide the available bandwidth to the proxy processes. Although specialized file systems may offer a significant improvement and overcome these limitations, we believe that user-level disk management on top of industry-standard file systems can offer similar performance advantages. In this paper, we study the overheads associated with file ...

Keywords: secondary storage, web caching, web performance, web proxies

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

THE ACM DIGITAL LIBRARY

 [Report a problem](#) [Satisfaction survey](#)

WEBCON: a toolkit for an automatic, data dictionary based connection of databases to the WWW

Full text Pdf (1.26 MB)

Source [Symposium on Applied Computing archive](#)
Proceedings of the 1998 ACM symposium on Applied Computing [table of contents](#)
 Atlanta, Georgia, United States
 Pages: 706 - 711
 Year of Publication: 1998
 ISBN:0-89791-969-6

Authors [Peter Zoller](#) Bavarian Research Centre for Knowledge-Based Systems (FORWISS), OrleansstraÙe 34, D-83667 Munich, Germany
[Ulrike Sommer](#) Bavarian Research Centre for Knowledge-Based Systems (FORWISS), OrleansstraÙe 34, D-83667 Munich, Germany

Sponsors [SIGADA](#): ACM Special Interest Group on Ada Programming Language
[SIGCUE](#): ACM Special Interest Group on Computer Uses In Education
[SIGAPP](#): ACM Special Interest Group on Applied Computing
[SIGBIO](#): ACM Special Interest Group on Biomedical Computing

Publisher ACM Press New York, NY, USA

Additional Information: [references](#) [index terms](#) [collaborative colleagues](#) [peer to peer](#)

Tools and Actions: [Find similar Articles](#) [Review this Article](#)
[Save this Article to a Binder](#) Display Formats: [BibTex](#) [EndNote](#) [ACM Ref](#)

DOI Bookmark: Use this link to bookmark this Article: <http://doi.acm.org/10.1145/330560.331071>
[What is a DOI?](#)

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

- 1 Allaire Corp.: Cold Fusion 3.0 Available at <http://www.allaire.com/>
- 2 Clausnitzer A.: Realisierung einer WWW- und einer ASCII-Version der OMNI\$.Recherche (Realization of a WWW and an ASCII version of the OMNIS search). Diplomathesis, TU Munich, Institute for Informatics, Dec. 15th 1994
- 3 EveryWare Development Inc.: Tango Enterprise 3.0 Available at <http://www.everyware.com/>
- 4 Henderson J.: Delivering Solutions for the Webcentric Enterprise. White paper. Available at <http://www.netdynamics.com/about/whitepaper.html>
- 5 Hunter A., Ferguson R.I., Hedges \$. SWOOP: An Application Generator for ORAC~ Systems. Fourth International Conference on the World Wide Web, Boston, 1-14 Dec. 1995. Available at

<http://www.w3.org/Confemmm,qWWW4>

6 IBM: Database 2 Wide Web Connection Version 1 Application Developer Guide. White Paper, 1996. Available at <http://www.software.ibm.com/data/db2/db2what.html>

7 Informix Software Inc.: informix Web DataBlade Module. Available at <http://www.informix.com/informix/products/techbrfs/dblade/datasht/webdb.html>

8 Netaway: SQL-Surfer available at <http://twww.netaway.com>

9 2 Technology: 02Web User Manual, Release 4.6. Jan. 1996

10 Oracle: Oracle Webserver 3.0. White Paper.

11 TransAction Software GmbH: Transbase Manuals Version 4.2.2. Munich, 1996

↑ INDEX TERMS

Primary Classification:

H. Information Systems

↳ H.3 INFORMATION STORAGE AND RETRIEVAL

↳ H.3.1 Content Analysis and Indexing

↳ **Subjects:** Dictionaries

Additional Classification:

H. Information Systems

↳ H.2 DATABASE MANAGEMENT

↳ H.2.4 Systems

↳ **Subjects:** Relational databases

↳ H.3 INFORMATION STORAGE AND RETRIEVAL

↳ H.3.5 On-line Information Services

↳ **Subjects:** Web-based services

↳ H.5 INFORMATION INTERFACES AND PRESENTATION (I.7)

↳ H.5.3 Group and Organization Interfaces

↳ **Subjects:** Web-based interaction

I. Computing Methodologies

↳ I.7 DOCUMENT AND TEXT PROCESSING

↳ I.7.2 Document Preparation

↳ **Nouns:** HTML

General Terms:

Design, Documentation, Human Factors, Languages, Management, Performance, Theory

Keywords:

[automatic page generation](#), [automatic query generation](#), [database to WWW connection](#), [dynamic page generation](#), [relational databases](#)

↑ **Collaborative Colleagues:**

Ulrike Sommer: Peter Zoller

Peter Zoller: Ulrike Sommer
 Günther Specht

↑ **Peer to Peer - Readers of this Article have also read:**

- [Data structures for quadtree approximation and compression](#) **Communications of the ACM** 28, 9
Hanan Samet
- [A hierarchical single-key-lock access control using the Chinese remainder theorem](#) **Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing**
Kim S. Lee , Huizhu Lu , D. D. Fisher
- [The GemStone object database management system](#) **Communications of the ACM** 34, 10
Paul Butterworth , Allen Otis , Jacob Stein
- [Putting innovation to work: adoption strategies for multimedia communication systems](#) **Communications of the ACM** 34, 12
Ellen Francik , Susan Ehrlich Rudman , Donna Cooper , Stephen Levine
- [An intelligent component database for behavioral synthesis](#) **Proceedings of the 27th ACM/IEEE conference on Design automation**
Gwo-Dong Chen , Daniel D. Gajski

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

[Sign in](#)

Google

[Web](#) [Images](#) [Video](#) [News](#) [Maps](#) [more »](#)[Advanced Search](#)
[Preferences](#)**Web**Results 101 - 110 of about 914,000 for **persistent web resources>1997**. (0.06 seconds)**Assessment Resource Center: General Education Skills Assessment ...**

Farmer, L. S. J. (1997). Authentic assessment of information literacy through electronic ...

Worries with the **Web**: A look at student use of **Web resources**. ...www.umuc.edu/odell/irahe/arc/4gen_inf.html - 33k - [Cached](#) - [Similar pages](#)**CS 395T: Web Operating Systems**What if the application needs to update **persistent** state? ... Manley, S., Seltzer, M., **Web**

Facts and Fantasy Proceedings of the 1997 USENIX Symposium on ...

www.cs.utexas.edu/~dahlin/Courses/**WebOS**/reading.html - 39k - [Cached](#) - [Similar pages](#)**Dublin Core Metadata in the RLG Information Landscape**

However, these issues have more immediate significance when the scope of the Dublin

Core is extended to non-**Web**-based **resources**. ...www.dlib.org/dlib/december97/12cromwell-kessler.html - 25k - [Cached](#) - [Similar pages](#)**On the Unification of Persistent Programming**Integrating a **persistent** system with the **Web** requires the management of an essential ...which allow better specification of **Web resources** than simple HTML. ...www.hippo.cis.strath.ac.uk/papers/unification.html - 55k - [Cached](#) - [Similar pages](#)**[PS] Web Proxy Caching: The Devil is in the Details**File Format: Adobe PostScript - [View as Text](#)**resources** that have cookies in. them. In the client trace described here, we ... USENIX,December 1997. [7] Jerrey Mogul. The case for **persistent**- ...www.cs.wisc.edu/~cao/WISP98/final-versions/anja.ps - [Similar pages](#)**[PDF] Persistent Identification of Electronic Documents and the Future ...**File Format: PDF/Adobe Acrobat - [View as HTML](#)**Persistent** Identification of **Web** Documents. ¶11. The ephemeral nature of Internet**resources** threatens to undermine the author- ...www.aallnet.org/products/pub_llj_v97n04/2005-42.pdf - [Similar pages](#)**Ten Design-less Rules for Successful Web Design**Ten Design-less Rules for Successful **Web** Design. <http://www.enosis.com/resources/10drules.html>. Nick Ragouzis, Enosis Group, January, 1997; June, 1998; ...www.enosis.com/**resources**/10drules.html - 8k - [Cached](#) - [Similar pages](#)**IETF WEBDAV Working Group Home Page****Web** Distributed Authoring and Versioning (WebDAV) Access Control Protocol ... creatingand maintaining a **persistent** ordering of the members of a collection. ...www.ics.uci.edu/~ejw/authoring/ - 18k - [Cached](#) - [Similar pages](#)**draft nottingham http roles 00 txt**6.1 **Persistent** Connections Servers MUST support a **persistent** connection if ... M.,"Examining the Cacheability of User- Requested **Web Resources**", 4th **Web** ...www.mnot.net/drafts/draft-nottingham-http-roles-00.txt - 15k - [Cached](#) - [Similar pages](#)**Cataloging Internet Resources: Organizing the Web in the Local ...**

Cataloging & Classification Quarterly, v. 25, no. 1 (1997): 11. 9 Jul., 93. ... "PURLs:

Persistent Uniform **Resource** Locators" PURL Home Page. ...

www.geocities.com/SoHo/Coffeehouse/3321/catweb.html - 35k - [Cached](#) - [Similar pages](#)

Result Page: **Previous** [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [19](#) [20](#) **Next**

[Search within results](#) | [Language Tools](#) | [Search Tips](#)

[Google Home](#) - [Advertising Programs](#) - [Business Solutions](#) - [About Google](#)

©2007 Google


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide

THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

 Terms used **composing internet searches**

 Found **28,250** of **198,310**

Sort results by


[Save results to a Binder](#)
[Try an Advanced Search](#)
[Try this search in The ACM Guide](#)

Display results


[Search Tips](#)
☐ Open results in a new window

Results 1 - 20 of 200

 Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale ☐ ☐ ☐ ☐ ☐

1 [Global digital museum: multimedia information access and creation on the Internet](#)


 Junichi Takahashi, Takayuki Kushida, Jung-Kook Hong, Shigeharu Sugita, Yasuyuki Kurita, Robert Rieger, Wendy Martin, Geri Gay, John Reeve, Rowena Loverance
 May 1998 **Proceedings of the third ACM conference on Digital libraries DL '98**

Publisher: ACM Press

Full text available: pdf(1.41 MB)

 Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

2 [A composable framework for secure multi-modal access to internet services from Post-PC devices](#)

Steven J. Ross, Jason L. Hill, Michael Y. Chen, Anthony D. Joseph, David E. Culler, Eric A. Brewer

 October 2002 **Mobile Networks and Applications**, Volume 7 Issue 5

Publisher: Kluwer Academic Publishers

Full text available: pdf(340.33 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The Post-PC revolution is bringing information access to a wide range of devices beyond the desktop, such as public kiosks, and mobile devices like cellular telephones, PDAs, and voice based vehicle telematics. However, existing deployed Internet services are geared toward the secure rich interface of private desktop computers. We propose the use of an infrastructure-based secure proxy architecture to bridge the gap between the capabilities of Post-PC devices and the requirements of Internet ser ...

Keywords: internet, middleware, post-PC, security, transcoding

3 [Effectively locating information on the Internet](#)



Qionsen Yu, Barrett R. Bryant

 April 1999 **Proceedings of the 37th annual Southeast regional conference (CD-ROM) ACM-SE 37**

Publisher: ACM Press

Full text available: pdf(25.46 KB)

 Additional Information: [full citation](#), [index terms](#)

4 [Constructing transliteration lexicons from web corpora](#)

Jin-Shea Kuo, Ying-Kuei Yang

July 2004 **Proceedings of the ACL 2004 on Interactive poster and demonstration sessions**

Publisher: Association for Computational Linguistics

Full text available:  [pdf\(138.97 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

This paper proposes a novel approach to automating the construction of transliterated-term lexicons. A simple syllable alignment algorithm is used to construct confusion matrices for cross-language syllable-phoneme conversion. Each row in the confusion matrix consists of a set of syllables in the source language that are (correctly or erroneously) matched phonetically and statistically to a syllable in the target language. Two conversions using phoneme-to-phoneme and text-to-phoneme syllabificat ...

5 Featured column: Computer literacy: essential in today's computer-centric world



Gireesh K. Gupta

June 2006 **ACM SIGCSE Bulletin**, Volume 38 Issue 2

Publisher: ACM Press

Full text available:  [pdf\(288.44 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Computer literacy is a fundamental part of undergraduate curriculum today. Computer literacy is as basic to undergraduate students as the course work in core curriculum in today's computer-centric information age [6]. The number of computers-in-use worldwide is growing, especially in the developing countries [3]. Computers affect every facet of our lives and every sector of the global society. Employers prefer workers who are computer literate because they are more productive and efficient at wo ...

Keywords: computer competency, computer literacy, digital divide

6 Extracting classification knowledge of Internet documents with mining term associations: a semantic approach



Shian-Hua Lin, Chi-Sheng Shih, Meng Chang Chen, Jan-Ming Ho, Ming-Tat Ko, Yueh-Ming Huang

August 1998 **Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '98**

Publisher: ACM Press


Full text available:  [pdf\(1.02 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

7 Placing search in context: the concept revisited



January 2002 **ACM Transactions on Information Systems (TOIS)**, Volume 20 Issue 1

Publisher: ACM Press

Full text available:  [pdf\(926.20 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Keyword-based search engines are in widespread use today as a popular means for Web-based information retrieval. Although such systems seem deceptively simple, a considerable amount of skill is required in order to satisfy non-trivial information needs. This paper presents a new conceptual paradigm for performing search in context, that largely automates the search process, providing even non-professional users with highly relevant results. This paradigm is implemented in practice in the Intelli ...

Keywords: Search, context, invisible web, semantic processing, statistical natural language processing

8 User evaluation of Físchlár-News: An automatic broadcast news delivery system



Hyowon Lee, Alan F. Smeaton, Noel E. O'Connor, Barry Smyth

April 2006 **ACM Transactions on Information Systems (TOIS)**, Volume 24 Issue 2

Publisher: ACM Press

Full text available: [pdf\(1.25 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Technological developments in content-based analysis of digital video information are undergoing much progress, with ideas for fully automatic systems now being proposed and demonstrated. Yet because we do not yet have robust operational video retrieval systems that can be deployed and used, the usual HCI practise of conducting a usage study and an informed iterative system design is thus not possible. Físchlár-News is one of the first automatic, content-based broadcast news analys ...

Keywords: User-evaluation, content-based video retrieval, usage analysis

9 Mobile web/accessibility overlaps: Evaluating interfaces for intelligent mobile search



Karen Church, Barry Smyth, Mark T. Keane

May 2006 **Proceedings of the 2006 international cross-disciplinary workshop on Web accessibility (W4A): Building the mobile web: rediscovering accessibility? W4A**

Publisher: ACM Press

Full text available: [pdf\(498.86 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Recent developments in the mobile phone market have led to a significant increase in the number of users accessing the Mobile Internet. Handsets have been improved to support a diverse range of content types (text, graphics, audio, video etc.), infrastructure investments have delivered improved bandwidth, and changes to billing models offer users much greater value for content. Today large numbers of users are moving away from browsing operator portals and towards off-portal search, leading to a ...

Keywords: mobile interfaces, mobile internet, mobile search, mobile web, search interfaces, user evaluation

10 Guidelines: Designing search engine user interfaces for the visually impaired



Barbara Leporini, Patrizia Andronico, Marina Buzzi

May 2004 **Proceedings of the 2004 international cross-disciplinary workshop on Web accessibility (W4A) W4A '04**

Publisher: ACM Press

Full text available: [pdf\(500.49 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Search engines are a fundamental tool for retrieving specific and appropriate information on the Internet; for this reason it is essential for any user to be able to interact with simple, clear and accessible interfaces. In this paper we describe the main design issues affecting the user interface of a search engine when a sightless user interacts by means of a screen reader or voice synthesizer. In particular, the most important differences between a visual layout and aural perception are discus ...

Keywords: Internet, accessibility, search engine, usability, user interface design, web navigation


11 A multilevel analysis of sociability, usability, and community dynamics in an online health community



Diane Maloney-Krichmar, Jenny Preece

June 2005 **ACM Transactions on Computer-Human Interaction (TOCHI)**, Volume 12 Issue

2

Publisher: ACM PressFull text available:  [pdf\(735.72 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

The aim of this research is to develop an in-depth understanding of the dynamics of online group interaction and the relationship between the participation in an online community and an individual's off-line life. The 2½-year study of a thriving online health support community (Bob's ACL WWWBoard) used a broad fieldwork approach, guided by the ethnographic research techniques of observation, interviewing, and archival research in combination with analysis of the group's dynamics during a on ...

Keywords: Online community, electronic support groups, health, online group dynamics, online social support, patient support community, self-help via the Internet, sociability, usability



12 A support system for revising titles to stimulate the lay reader's interest in technical achievements

Yasuko Senda, Yasusi Sinohara, Manabu Okumura

August 2004 **Proceedings of the 20th international conference on Computational Linguistics COLING '04****Publisher:** Association for Computational LinguisticsFull text available:  [pdf\(251.51 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

When we write a report or an explanation on a newly-developed technology for readers including laypersons, it is very important to compose a title that can stimulate their interest in the technology. However, it is difficult for inexperienced authors to come up with an appealing title. In this research, we developed a support system for revising titles. We call it "title revision wizard". The wizard provides a guidance on revising draft title to compose a title meeting three key points, and suppo ...

13 Web engineering: Evaluation of crawling policies for a web-repository crawler


 Frank McCown, Michael L. NelsonAugust 2006 **Proceedings of the seventeenth conference on Hypertext and hypermedia HYPERTEXT '06****Publisher:** ACM PressFull text available:  [pdf\(482.40 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We have developed a web-repository crawler that is used for reconstructing websites when backups are unavailable. Our crawler retrieves web resources from the Internet Archive, Google, Yahoo and MSN. We examine the challenges of crawling web repositories, and we discuss strategies for overcoming some of these obstacles. We propose three crawling policies which can be used to reconstruct websites. We evaluate the effectiveness of the policies by reconstructing 24 websites and comparing the result ...

Keywords: crawler policy, digital preservation, search engine, website reconstruction

14 Interactive Internet search: keyword, directory and query reformulation mechanisms compared

 Peter Bruza, Robert McArthur, Simon DennisJuly 2000 **Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval SIGIR '00****Publisher:** ACM Press

Full text available:  pdf(1.05 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This article compares search effectiveness when using query-based Internet search (via the Google search engine), directory-based search (via Yahoo) and phrase-based query reformulation assisted search (via the Hyperindex browser) by means of a controlled, user-based experimental study. The focus was to evaluate aspects of the search process. Cognitive load was measured using a secondary digit-monitoring task to quantify the effort of the user in various search states; independent relevance j ...

Keywords: field/slash;empirical studies of the information seeking pro, monitoring user behaviour to improve search, navigation versus ad hoc search, testing methodology

15 Full papers: Losers and finders: indexing audio-visual digital media



Mike Leggett

April 2005 **Proceedings of the 5th conference on Creativity & cognition C&C '05**

Publisher: ACM Press

Full text available:  pdf(323.17 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The contemporary burgeoning usage of digital movies, photos, audio and text, their distribution through networks both electronic and physical will be considered in the context of a convergence of these media with a popular interest in personal and community history and identity. The paper introduces interdisciplinary research into human memory as a context for understanding its relation to machine memory and methods of storing and retrieval. It proposes an approach to indexing audio-visual media ...

Keywords: digital media, index, interactive, taxonomy

16 Pervasive computing: what is it good for?



Andrew C. Huang, Benjamin C. Ling, Shankar Ponnekanti

August 1999 **Proceedings of the 1st ACM international workshop on Data engineering for wireless and mobile access MobiDe '99**

Publisher: ACM Press


Full text available:  pdf(897.82 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

17 Distributed information retrieval with skewed database size distributions

Luo Si, Jie Lu, Jamie Callan

May 2003 **Proceedings of the 2003 annual national conference on Digital government research dg.o '03**

Publisher: Digital Government Research Center

Full text available:  pdf(69.64 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)


The proliferation of government information on local area networks and the Internet creates the problem of finding information that may be distributed among many disjoint text databases (*distributed information retrieval* or *federated search*). A distributed information retrieval system is composed of three components: Resource representation, resource selection and result merging. Previous research suggested that the CORI algorithm is one of the most effective resource selection alg ...

18 Session 6A: Architectures: The evolution of software evolvability



Chris Lürer, David S. Rosenblum, André van der Hoek

September 2001 **Proceedings of the 4th International Workshop on Principles of**

Software Evolution IWPSE '01**Publisher:** ACM PressFull text available:  [pdf\(392.75 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#)

We analyze two trends that have influenced the evolvability of component-based applications: increase of component exchangeability and increase of component distance. Exchangeability mechanisms can be classified either as code reuse or as service reuse. Component distance can vary from file scope to Internet scope. We discuss the various stages of evolvability in these dimensions, describe the state of the art, and speculate on future developments.

Keywords: code reuse, evolvability, exchangeability, service reuse**19 Pervasive Servers: A framework for creating a society of appliances**


Tatsuo Nakajima

July 2003 **Personal and Ubiquitous Computing**, Volume 7 Issue 3-4**Publisher:** Springer-VerlagFull text available:  [pdf\(352.36 KB\)](#) Additional Information: [full citation](#), [abstract](#), [index terms](#)

The paper proposes a framework to support spontaneous interaction among information appliances in our daily computing environments by creating a society of appliances. Our framework, that we call Pervasive Servers, embeds micro-servers called stationary pervasive servers everywhere. Also, a personal pervasive server that is carried by each person coordinates the embedded servers that are near to the person. The framework is very attractive because it is easy to personalise the coordination accor ...

Keywords: Appliances, Software infrastructure, Spontaneous interaction**20 User-level internet path diagnosis**

Ratul Mahajan, Neil Spring, David Wetherall, Thomas Anderson

October 2003 **ACM SIGOPS Operating Systems Review , Proceedings of the nineteenth ACM symposium on Operating systems principles SOSP '03**, Volume 37 Issue 5**Publisher:** ACM PressFull text available:  [pdf\(403.57 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citing's](#), [index terms](#)





Diagnosing faults in the Internet is arduous and time-consuming, in part because the network is composed of diverse components spread across many administrative domains. We consider an extreme form of this problem: can end users, with no special privileges, identify and pinpoint faults inside the network that degrade the performance of their applications? To answer this question, we present both an architecture for user-level Internet path diagnosis and a practical tool to diagnose paths in the ...

Keywords: measurement tools, path diagnosis

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Network Working Group
Request for Comments: 2396
Updates: 1808, 1738
Category: Standards Track

T. Berners-Lee
MIT/LCS
R. Fielding
U.C. Irvine
L. Masinter
Xerox Corporation
August 1998

Uniform Resource Identifiers (URI): Generic Syntax

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1998). All Rights Reserved.

IESG Note

This paper describes a "superset" of operations that can be applied to URI. It consists of both a grammar and a description of basic functionality for URI. To understand what is a valid URI, both the grammar and the associated description have to be studied. Some of the functionality described is not applicable to all URI schemes, and some operations are only possible when certain media types are retrieved using the URI, regardless of the scheme used.

Abstract

A Uniform Resource Identifier (URI) is a compact string of characters for identifying an abstract or physical resource. This document defines the generic syntax of URI, including both absolute and relative forms, and guidelines for their use; it revises and replaces the generic definitions in RFC 1738 and RFC 1808.

This document defines a grammar that is a superset of all valid URI, such that an implementation can parse the common components of a URI reference without knowing the scheme-specific requirements of every possible identifier type. This document does not define a generative grammar for URI; that task will be performed by the individual specifications of each URI scheme.

Berners-Lee, et. al.

Standards Track

[Page 1]

□

RFC 2396

URI Generic Syntax

August 1998

1. Introduction

Uniform Resource Identifiers (URI) provide a simple and extensible means for identifying a resource. This specification of URI syntax and semantics is derived from concepts introduced by the World Wide Web global information initiative, whose use of such objects dates from 1990 and is described in "Universal Resource Identifiers in WWW" [RFC1630]. The specification of URI is designed to meet the recommendations laid out in "Functional Recommendations for Internet Resource Locators" [RFC1736] and "Functional Requirements for Uniform Resource Names" [RFC1737].

This document updates and merges "Uniform Resource Locators" [RFC1738] and "Relative Uniform Resource Locators" [RFC1808] in order to define a single, generic syntax for all URI. It excludes those portions of RFC 1738 that defined the specific syntax of individual URL schemes; those portions will be updated as separate documents, as will the process for registration of new URI schemes. This document does not discuss the issues and recommendation for dealing with characters outside of the US-ASCII character set [ASCII]; those recommendations are discussed in a separate document.

All significant changes from the prior RFCs are noted in Appendix G.

1.1 Overview of URI

URI are characterized by the following definitions:

Uniform

Uniformity provides several benefits: it allows different types of resource identifiers to be used in the same context, even when the mechanisms used to access those resources may differ; it allows uniform semantic interpretation of common syntactic conventions across different types of resource identifiers; it allows introduction of new types of resource identifiers without interfering with the way that existing identifiers are used; and, it allows the identifiers to be reused in many different contexts, thus permitting new applications or protocols to leverage a pre-existing, large, and widely-used set of resource identifiers.

Resource

A resource can be anything that has identity. Familiar examples include an electronic document, an image, a service (e.g., "today's weather report for Los Angeles"), and a collection of other resources. Not all resources are network "retrievable"; e.g., human beings, corporations, and bound books in a library can also be considered resources.

Berners-Lee, et. al.

Standards Track

[Page 2]

□

RFC 2396

URI Generic Syntax

August 1998

The resource is the conceptual mapping to an entity or set of entities, not necessarily the entity which corresponds to that mapping at any particular instance in time. Thus, a resource can remain constant even when its content---the entities to

which it currently corresponds---changes over time, provided that the conceptual mapping is not changed in the process.

Identifier

An identifier is an object that can act as a reference to something that has identity. In the case of URI, the object is a sequence of characters with a restricted syntax.

Having identified a resource, a system may perform a variety of operations on the resource, as might be characterized by such words as 'access', 'update', 'replace', or 'find attributes'.

1.2. URI, URL, and URN

A URI can be further classified as a locator, a name, or both. The term "Uniform Resource Locator" (URL) refers to the subset of URI that identify resources via a representation of their primary access mechanism (e.g., their network "location"), rather than identifying the resource by name or by some other attribute(s) of that resource. The term "Uniform Resource Name" (URN) refers to the subset of URI that are required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable.

The URI scheme (Section 3.1) defines the namespace of the URI, and thus may further restrict the syntax and semantics of identifiers using that scheme. This specification defines those elements of the URI syntax that are either required of all URI schemes or are common to many URI schemes. It thus defines the syntax and semantics that are needed to implement a scheme-independent parsing mechanism for URI references, such that the scheme-dependent handling of a URI can be postponed until the scheme-dependent semantics are needed. We use the term URL below when describing syntax or semantics that only apply to locators.

Although many URL schemes are named after protocols, this does not imply that the only way to access the URL's resource is via the named protocol. Gateways, proxies, caches, and name resolution services might be used to access some resources, independent of the protocol of their origin, and the resolution of some URL may require the use of more than one protocol (e.g., both DNS and HTTP are typically used to access an "http" URL's resource when it can't be found in a local cache).

A URN differs from a URL in that it's primary purpose is persistent labeling of a resource with an identifier. That identifier is drawn from one of a set of defined namespaces, each of which has its own set name structure and assignment procedures. The "urn" scheme has been reserved to establish the requirements for a standardized URN namespace, as defined in "URN Syntax" [RFC2141] and its related specifications.

Most of the examples in this specification demonstrate URL, since they allow the most varied use of the syntax and often have a hierarchical namespace. A parser of the URI syntax is capable of parsing both URL and URN references as a generic URI; once the scheme is determined, the scheme-specific parsing can be performed on the generic URI components. In other words, the URI syntax is a superset of the syntax of all URI schemes.

1.3. Example URI

The following examples illustrate URI that are in common use.

```
ftp://ftp.is.co.za/rfc/rfc1808.txt
-- ftp scheme for File Transfer Protocol services

gopher://spinaltap.micro.umn.edu/00/Weather/California/Los%20Angeles
-- gopher scheme for Gopher and Gopher+ Protocol services

http://www.math.uio.no/faq/compression-faq/part1.html
-- http scheme for Hypertext Transfer Protocol services

mailto:mduerst@ifi.unizh.ch
-- mailto scheme for electronic mail addresses

news:comp.infosystems.www.servers.unix
-- news scheme for USENET news groups and articles

telnet://melvyl.ucop.edu/
-- telnet scheme for interactive services via the TELNET Protocol
```

1.4. Hierarchical URI and Relative Forms

An absolute identifier refers to a resource independent of the context in which the identifier is used. In contrast, a relative identifier refers to a resource by describing the difference within a hierarchical namespace between the current context and an absolute identifier of the resource.

Berners-Lee, et. al.	Standards Track	[Page 4]
□		
RFC 2396	URI Generic Syntax	August 1998

Some URI schemes support a hierarchical naming system, where the hierarchy of the name is denoted by a "/" delimiter separating the components in the scheme. This document defines a scheme-independent 'relative' form of URI reference that can be used in conjunction with a 'base' URI (of a hierarchical scheme) to produce another URI. The syntax of hierarchical URI is described in Section 3; the relative URI calculation is described in Section 5.

1.5. URI Transcribability

The URI syntax was designed with global transcribability as one of its main concerns. A URI is a sequence of characters from a very

limited set, i.e. the letters of the basic Latin alphabet, digits, and a few special characters. A URI may be represented in a variety of ways: e.g., ink on paper, pixels on a screen, or a sequence of octets in a coded character set. The interpretation of a URI depends only on the characters used and not how those characters are represented in a network protocol.

The goal of transcribability can be described by a simple scenario. Imagine two colleagues, Sam and Kim, sitting in a pub at an international conference and exchanging research ideas. Sam asks Kim for a location to get more information, so Kim writes the URI for the research site on a napkin. Upon returning home, Sam takes out the napkin and types the URI into a computer, which then retrieves the information to which Kim referred.

There are several design concerns revealed by the scenario:

- o A URI is a sequence of characters, which is not always represented as a sequence of octets.
- o A URI may be transcribed from a non-network source, and thus should consist of characters that are most likely to be able to be typed into a computer, within the constraints imposed by keyboards (and related input devices) across languages and locales.
- o A URI often needs to be remembered by people, and it is easier for people to remember a URI when it consists of meaningful components.

These design concerns are not always in alignment. For example, it is often the case that the most meaningful name for a URI component would require characters that cannot be typed into some systems. The ability to transcribe the resource identifier from one medium to another was considered more important than having its URI consist of the most meaningful of components. In local and regional contexts

and with improving technology, users might benefit from being able to use a wider range of characters; such use is not defined in this document.

1.6. Syntax Notation and Common Elements

This document uses two conventions to describe and define the syntax for URI. The first, called the layout form, is a general description of the order of components and component separators, as in

```
<first>/<second>;<third>?<fourth>
```

The component names are enclosed in angle-brackets and any characters outside angle-brackets are literal separators. Whitespace should be ignored. These descriptions are used informally and do not define the syntax requirements.

The second convention is a BNF-like grammar, used to define the formal URI syntax. The grammar is that of [RFC822], except that "|" is used to designate alternatives. Briefly, rules are separated from definitions by an equal "=", indentation is used to continue a rule definition over more than one line, literals are quoted with "", parentheses "(" and ")" are used to group elements, optional elements are enclosed in "[" and "]" brackets, and elements may be preceded with <n>* to designate n or more repetitions of the following element; n defaults to 0.

Unlike many specifications that use a BNF-like grammar to define the bytes (octets) allowed by a protocol, the URI grammar is defined in terms of characters. Each literal in the grammar corresponds to the character it represents, rather than to the octet encoding of that character in any particular coded character set. How a URI is represented in terms of bits and bytes on the wire is dependent upon the character encoding of the protocol used to transport it, or the charset of the document which contains it.

The following definitions are common to many elements:

alpha = lowalpha | upalpha

lowalpha = "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" |
 "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" |
 "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z"

upalpha = "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" |
 "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
 "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"

Berners-Lee, et. al.

Standards Track

[Page 6]

□

RFC 2396

URI Generic Syntax

August 1998

digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" |
 "8" | "9"

alphanum = alpha | digit

The complete URI syntax is collected in Appendix A.

2. URI Characters and Escape Sequences

URI consist of a restricted set of characters, primarily chosen to aid transcribability and usability both in computer systems and in non-computer communications. Characters used conventionally as delimiters around URI were excluded. The restricted set of characters consists of digits, letters, and a few graphic symbols were chosen from those common to most of the character encodings and input facilities available to Internet users.

uric = reserved | unreserved | escaped

Within a URI, characters are either used as delimiters, or to

represent strings of data (octets) within the delimited portions. Octets are either represented directly by a character (using the US-ASCII character for that octet [ASCII]) or by an escape encoding. This representation is elaborated below.

2.1 URI and non-ASCII characters

The relationship between URI and characters has been a source of confusion for characters that are not part of US-ASCII. To describe the relationship, it is useful to distinguish between a "character" (as a distinguishable semantic entity) and an "octet" (an 8-bit byte). There are two mappings, one from URI characters to octets, and a second from octets to original characters:

URI character sequence->octet sequence->original character sequence

A URI is represented as a sequence of characters, not as a sequence of octets. That is because URI might be "transported" by means that are not through a computer network, e.g., printed on paper, read over the radio, etc.

A URI scheme may define a mapping from URI characters to octets; whether this is done depends on the scheme. Commonly, within a delimited component of a URI, a sequence of characters may be used to represent a sequence of octets. For example, the character "a" represents the octet 97 (decimal), while the character sequence "%", "0", "a" represents the octet 10 (decimal).

Berners-Lee, et. al.

Standards Track

[Page 7]

□

RFC 2396

URI Generic Syntax

August 1998

There is a second translation for some resources: the sequence of octets defined by a component of the URI is subsequently used to represent a sequence of characters. A 'charset' defines this mapping. There are many charsets in use in Internet protocols. For example, UTF-8 [UTF-8] defines a mapping from sequences of octets to sequences of characters in the repertoire of ISO 10646.

In the simplest case, the original character sequence contains only characters that are defined in US-ASCII, and the two levels of mapping are simple and easily invertible: each 'original character' is represented as the octet for the US-ASCII code for it, which is, in turn, represented as either the US-ASCII character, or else the "%" escape sequence for that octet.

For original character sequences that contain non-ASCII characters, however, the situation is more difficult. Internet protocols that transmit octet sequences intended to represent character sequences are expected to provide some way of identifying the charset used, if there might be more than one [RFC2277]. However, there is currently no provision within the generic URI syntax to accomplish this identification. An individual URI scheme may require a single charset, define a default charset, or provide a way to indicate the charset used.

It is expected that a systematic treatment of character encoding within URI will be developed as a future modification of this specification.

2.2. Reserved Characters

Many URI include components consisting of or delimited by, certain special characters. These characters are called "reserved", since their usage within the URI component is limited to their reserved purpose. If the data for a URI component would conflict with the reserved purpose, then the conflicting data must be escaped before forming the URI.

```
reserved    = ";" | "/" | "?" | ":" | "@" | "&" | "=" | "+" |
              "$" | ","
```

The "reserved" syntax class above refers to those characters that are allowed within a URI, but which may not be allowed within a particular component of the generic URI syntax; they are used as delimiters of the components described in Section 3.

Berners-Lee, et. al.

Standards Track

[Page 8]

□

RFC 2396

URI Generic Syntax

August 1998

Characters in the "reserved" set are not reserved in all contexts. The set of characters actually reserved within any given URI component is defined by that component. In general, a character is reserved if the semantics of the URI changes if the character is replaced with its escaped US-ASCII encoding.

2.3. Unreserved Characters

Data characters that are allowed in a URI but do not have a reserved purpose are called unreserved. These include upper and lower case letters, decimal digits, and a limited set of punctuation marks and symbols.

```
unreserved  = alphanum | mark
```

```
mark        = "-" | "_" | "." | "!" | "~" | "*" | "'" | "(" | ")"
```

Unreserved characters can be escaped without changing the semantics of the URI, but this should not be done unless the URI is being used in a context that does not allow the unescaped character to appear.

2.4. Escape Sequences

Data must be escaped if it does not have a representation using an unreserved character; this includes data that does not correspond to a printable character of the US-ASCII coded character set, or that corresponds to any US-ASCII character that is disallowed, as explained below.

2.4.1. Escaped Encoding

An escaped octet is encoded as a character triplet, consisting of the percent character "%" followed by the two hexadecimal digits representing the octet code. For example, "%20" is the escaped encoding for the US-ASCII space character.

```

escaped    = "%" hex hex
hex        = digit | "A" | "B" | "C" | "D" | "E" | "F" |
              "a" | "b" | "c" | "d" | "e" | "f"

```

2.4.2. When to Escape and Unescape

A URI is always in an "escaped" form, since escaping or unescaping a completed URI might change its semantics. Normally, the only time escape encodings can safely be made is when the URI is being created from its component parts; each component may have its own set of characters that are reserved, so only the mechanism responsible for generating or interpreting that component can determine whether or

Berners-Lee, et. al.

Standards Track

[Page 9]

□

RFC 2396

URI Generic Syntax

August 1998

not escaping a character will change its semantics. Likewise, a URI must be separated into its components before the escaped characters within those components can be safely decoded.

In some cases, data that could be represented by an unreserved character may appear escaped; for example, some of the unreserved "mark" characters are automatically escaped by some systems. If the given URI scheme defines a canonicalization algorithm, then unreserved characters may be unescaped according to that algorithm. For example, "%7e" is sometimes used instead of "~" in an http URL path, but the two are equivalent for an http URL.

Because the percent "%" character always has the reserved purpose of being the escape indicator, it must be escaped as "%25" in order to be used as data within a URI. Implementers should be careful not to escape or unescape the same string more than once, since unescaping an already unescaped string might lead to misinterpreting a percent data character as another escaped character, or vice versa in the case of escaping an already escaped string.

2.4.3. Excluded US-ASCII Characters

Although they are disallowed within the URI syntax, we include here a description of those US-ASCII characters that have been excluded and the reasons for their exclusion.

The control characters in the US-ASCII coded character set are not used within a URI, both because they are non-printable and because they are likely to be misinterpreted by some control mechanisms.

```
control    = <US-ASCII coded characters 00-1F and 7F hexadecimal>
```

The space character is excluded because significant spaces may disappear and insignificant spaces may be introduced when URI are transcribed or typeset or subjected to the treatment of word-processing programs. Whitespace is also used to delimit URI in many contexts.

space = <US-ASCII coded character 20 hexadecimal>

The angle-bracket "<" and ">" and double-quote (") characters are excluded because they are often used as the delimiters around URI in text documents and protocol fields. The character "#" is excluded because it is used to delimit a URI from a fragment identifier in URI references (Section 4). The percent character "%" is excluded because it is used for the encoding of escaped characters.

delims = "<" | ">" | "#" | "%" | "<"

Berners-Lee, et. al.	Standards Track	[Page 10]
□		
RFC 2396	URI Generic Syntax	August 1998

Other characters are excluded because gateways and other transport agents are known to sometimes modify such characters, or they are used as delimiters.

unwise = "{" | "}" | "|" | "\" | "^" | "[" | "]" | "`"

Data corresponding to excluded characters must be escaped in order to be properly represented within a URI.

3. URI Syntactic Components

The URI syntax is dependent upon the scheme. In general, absolute URI are written as follows:

<scheme>:<scheme-specific-part>

An absolute URI contains the name of the scheme being used (<scheme>) followed by a colon (":") and then a string (the <scheme-specific-part>) whose interpretation depends on the scheme.

The URI syntax does not require that the scheme-specific-part have any general structure or set of semantics which is common among all URI. However, a subset of URI do share a common syntax for representing hierarchical relationships within the namespace. This "generic URI" syntax consists of a sequence of four main components:

<scheme>://<authority><path>?<query>

each of which, except <scheme>, may be absent from a particular URI. For example, some URI schemes do not allow an <authority> component, and others do not use a <query> component.

absoluteURI = scheme ":" (hier_part | opaque_part)

URI that are hierarchical in nature use the slash "/" character for separating hierarchical components. For some file systems, a "/"

character (used to denote the hierarchical structure of a URI) is the delimiter used to construct a file name hierarchy, and thus the URI path will look similar to a file pathname. This does NOT imply that the resource is a file or that the URI maps to an actual filesystem pathname.

```
hier_part      = ( net_path | abs_path ) [ "?" query ]
net_path       = "://" authority [ abs_path ]
abs_path       = "/" path_segments
```

Berners-Lee, et. al.	Standards Track	[Page 11]
□		
RFC 2396	URI Generic Syntax	August 1998

URI that do not make use of the slash "/" character for separating hierarchical components are considered opaque by the generic URI parser.

```
opaque_part    = uric_no_slash *uric
uric_no_slash  = unreserved | escaped | ";" | "?" | ":" | "@" |
                "&" | "=" | "+" | "$" | ","
```

We use the term <path> to refer to both the <abs_path> and <opaque_part> constructs, since they are mutually exclusive for any given URI and can be parsed as a single component.

3.1. Scheme Component

Just as there are many different methods of access to resources, there are a variety of schemes for identifying such resources. The URI syntax consists of a sequence of components separated by reserved characters, with the first component defining the semantics for the remainder of the URI string.

Scheme names consist of a sequence of characters beginning with a lower case letter and followed by any combination of lower case letters, digits, plus ("+"), period ("."), or hyphen ("-"). For resiliency, programs interpreting URI should treat upper case letters as equivalent to lower case in scheme names (e.g., allow "HTTP" as well as "http").

```
scheme         = alpha *( alpha | digit | "+" | "-" | "." )
```

Relative URI references are distinguished from absolute URI in that they do not begin with a scheme name. Instead, the scheme is inherited from the base URI, as described in Section 5.2.

3.2. Authority Component

Many URI schemes include a top hierarchical element for a naming authority, such that the namespace defined by the remainder of the URI is governed by that authority. This authority component is typically defined by an Internet-based server or a scheme-specific

registry of naming authorities.

authority = server | reg_name

The authority component is preceded by a double slash "/" and is terminated by the next slash "/", question-mark "?", or by the end of the URI. Within the authority component, the characters ";", ":", "@", "?", and "/" are reserved.

Berners-Lee, et. al.	Standards Track	[Page 12]
□		
RFC 2396	URI Generic Syntax	August 1998

An authority component is not required for a URI scheme to make use of relative references. A base URI without an authority component implies that any relative reference will also be without an authority component.

3.2.1. Registry-based Naming Authority

The structure of a registry-based naming authority is specific to the URI scheme, but constrained to the allowed characters for an authority component.

reg_name = 1*(unreserved | escaped | "\$" | "," |
";" | ":" | "@" | "&" | "=" | "+")

3.2.2. Server-based Naming Authority

URL schemes that involve the direct use of an IP-based protocol to a specified server on the Internet use a common syntax for the server component of the URI's scheme-specific data:

<userinfo>@<host>:<port>

where <userinfo> may consist of a user name and, optionally, scheme-specific information about how to gain authorization to access the server. The parts "<userinfo>@" and ":<port>" may be omitted.

server = [[userinfo "@"] hostport]

The user information, if present, is followed by a commercial at-sign "@".

userinfo = *(unreserved | escaped |
";" | ":" | "&" | "=" | "+" | "\$" | ",")

Some URL schemes use the format "user:password" in the userinfo field. This practice is NOT RECOMMENDED, because the passing of authentication information in clear text (such as URI) has proven to be a security risk in almost every case where it has been used.

The host is a domain name of a network host, or its IPv4 address as a set of four decimal digit groups separated by ".". Literal IPv6 addresses are not supported.

hostport = host [":" port]

```

host           = hostname | IPv4address
hostname       = *( domainlabel "." ) toplabel [ "." ]
domainlabel    = alphanum | alphanum *( alphanum | "-" ) alphanum
toplabel       = alpha | alpha *( alphanum | "-" ) alphanum

```

Berners-Lee, et. al. Standards Track [Page 13]
 □
 RFC 2396 URI Generic Syntax August 1998

```

IPv4address    = 1*digit "." 1*digit "." 1*digit "." 1*digit
port           = *digit

```

Hostnames take the form described in Section 3 of [RFC1034] and Section 2.1 of [RFC1123]: a sequence of domain labels separated by ".", each domain label starting and ending with an alphanumeric character and possibly also containing "-" characters. The rightmost domain label of a fully qualified domain name will never start with a digit, thus syntactically distinguishing domain names from IPv4 addresses, and may be followed by a single "." if it is necessary to distinguish between the complete domain name and any local domain. To actually be "Uniform" as a resource locator, a URL hostname should be a fully qualified domain name. In practice, however, the host component may be a local domain literal.

Note: A suitable representation for including a literal IPv6 address as the host part of a URL is desired, but has not yet been determined or implemented in practice.

The port is the network port number for the server. Most schemes designate protocols that have a default port number. Another port number may optionally be supplied, in decimal, separated from the host by a colon. If the port is omitted, the default port number is assumed.

3.3. Path Component

The path component contains data, specific to the authority (or the scheme if there is no authority component), identifying the resource within the scope of that scheme and authority.

```

path           = [ abs_path | opaque_part ]

path_segments  = segment *( "/" segment )
segment        = *pchar *( ";" param )
param          = *pchar

pchar          = unreserved | escaped |
                 ":" | "@" | "&" | "=" | "+" | "$" | ","

```

The path may consist of a sequence of path segments separated by a single slash "/" character. Within a path segment, the characters "/", ";", "=", and "?" are reserved. Each path segment may include a sequence of parameters, indicated by the semicolon ";" character. The parameters are not significant to the parsing of relative references.

3.4. Query Component

The query component is a string of information to be interpreted by the resource.

query = *uric

Within a query component, the characters ";", "/", "?", ":", "@", "&", "=", "+", ",", and "\$" are reserved.

4. URI References

The term "URI-reference" is used here to denote the common usage of a resource identifier. A URI reference may be absolute or relative, and may have additional information attached in the form of a fragment identifier. However, "the URI" that results from such a reference includes only the absolute URI after the fragment identifier (if any) is removed and after any relative URI is resolved to its absolute form. Although it is possible to limit the discussion of URI syntax and semantics to that of the absolute result, most usage of URI is within general URI references, and it is impossible to obtain the URI from such a reference without also parsing the fragment and resolving the relative form.

URI-reference = [absoluteURI | relativeURI] ["#" fragment]

The syntax for relative URI is a shortened form of that for absolute URI, where some prefix of the URI is missing and certain path components ("." and "..") have a special meaning when, and only when, interpreting a relative path. The relative URI syntax is defined in Section 5.

4.1. Fragment Identifier

When a URI reference is used to perform a retrieval action on the identified resource, the optional fragment identifier, separated from the URI by a crosshatch ("#") character, consists of additional reference information to be interpreted by the user agent after the retrieval action has been successfully completed. As such, it is not part of a URI, but is often used in conjunction with a URI.

fragment = *uric

The semantics of a fragment identifier is a property of the data resulting from a retrieval action, regardless of the type of URI used in the reference. Therefore, the format and interpretation of fragment identifiers is dependent on the media type [RFC2046] of the retrieval result. The character restrictions described in Section 2

□

for URI also apply to the fragment in a URI-reference. Individual media types may define additional restrictions or structure within the fragment for specifying different types of "partial views" that can be identified within that media type.

A fragment identifier is only meaningful when a URI reference is intended for retrieval and the result of that retrieval is a document for which the identified fragment is consistently defined.

4.2. Same-document References

A URI reference that does not contain a URI is a reference to the current document. In other words, an empty URI reference within a document is interpreted as a reference to the start of that document, and a reference containing only a fragment identifier is a reference to the identified fragment of that document. Traversal of such a reference should not result in an additional retrieval action. However, if the URI reference occurs in a context that is always intended to result in a new request, as in the case of HTML's FORM element, then an empty URI reference represents the base URI of the current document and should be replaced by that URI when transformed into a request.

4.3. Parsing a URI Reference

A URI reference is typically parsed according to the four main components and fragment identifier in order to determine what components are present and whether the reference is relative or absolute. The individual components are then parsed for their subparts and, if not opaque, to verify their validity.

Although the BNF defines what is allowed in each component, it is ambiguous in terms of differentiating between an authority component and a path component that begins with two slash characters. The greedy algorithm is used for disambiguation: the left-most matching rule soaks up as much of the URI reference string as it is capable of matching. In other words, the authority component wins.

Readers familiar with regular expressions should see Appendix B for a concrete parsing example and test oracle.

5. Relative URI References

It is often the case that a group or "tree" of documents has been constructed to serve a common purpose; the vast majority of URI in these documents point to resources within the tree rather than

□

outside of it. Similarly, documents located at a particular site are much more likely to refer to other resources at that site than to resources at remote sites.

Relative addressing of URI allows document trees to be partially independent of their location and access scheme. For instance, it is possible for a single set of hypertext documents to be simultaneously accessible and traversable via each of the "file", "http", and "ftp" schemes if the documents refer to each other using relative URI. Furthermore, such document trees can be moved, as a whole, without changing any of the relative references. Experience within the WWW has demonstrated that the ability to perform relative referencing is necessary for the long-term usability of embedded URI.

The syntax for relative URI takes advantage of the <hier_part> syntax of <absoluteURI> (Section 3) in order to express a reference that is relative to the namespace of another hierarchical URI.

```
relativeURI = ( net_path | abs_path | rel_path ) [ "?" query ]
```

A relative reference beginning with two slash characters is termed a network-path reference, as defined by <net_path> in Section 3. Such references are rarely used.

A relative reference beginning with a single slash character is termed an absolute-path reference, as defined by <abs_path> in Section 3.

A relative reference that does not begin with a scheme name or a slash character is termed a relative-path reference.

```
rel_path = rel_segment [ abs_path ]
```

```
rel_segment = 1*( unreserved | escaped |
                  ";" | "@" | "&" | "=" | "+" | "$" | "," )
```

Within a relative-path reference, the complete path segments "." and ".." have special meanings: "the current hierarchy level" and "the level above this hierarchy level", respectively. Although this is very similar to their use within Unix-based filesystems to indicate directory levels, these path components are only considered special when resolving a relative-path reference to its absolute form (Section 5.2).

Authors should be aware that a path segment which contains a colon character cannot be used as the first segment of a relative URI path (e.g., "this:that"), because it would be mistaken for a scheme name.

It is therefore necessary to precede such segments with other segments (e.g., "./this:that") in order for them to be referenced as a relative path.

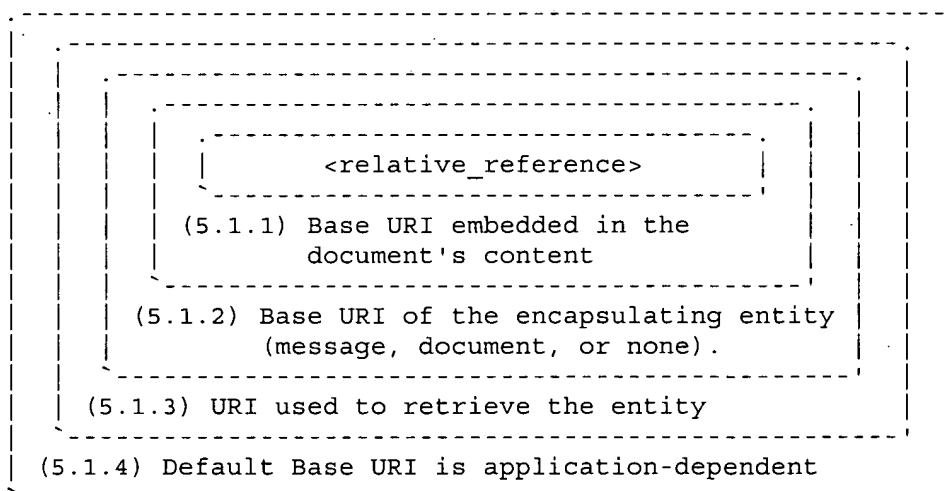
It is not necessary for all URI within a given scheme to be restricted to the <hier_part> syntax, since the hierarchical properties of that syntax are only necessary when relative URI are used within a particular document. Documents can only make use of relative URI when their base URI fits within the <hier_part> syntax. It is assumed that any document which contains a relative reference will also have a base URI that obeys the syntax. In other words, relative URI cannot be used within a document that has an unsuitable base URI.

Some URI schemes do not allow a hierarchical syntax matching the <hier_part> syntax, and thus cannot use relative references.

5.1. Establishing a Base URI

The term "relative URI" implies that there exists some absolute "base URI" against which the relative reference is applied. Indeed, the base URI is necessary to define the semantics of any relative URI reference; without it, a relative reference is meaningless. In order for relative URI to be usable within a document, the base URI of that document must be known to the parser.

The base URI of a document can be established in one of four ways, listed below in order of precedence. The order of precedence can be thought of in terms of layers, where the innermost defined base URI has the highest precedence. This can be visualized graphically as:



5.1.1. Base URI within Document Content

Within certain document media types, the base URI of the document can be embedded within the content itself such that it can be readily obtained by a parser. This can be useful for descriptive documents, such as tables of content, which may be transmitted to others through protocols other than their usual retrieval context (e.g., E-Mail or USENET news).

It is beyond the scope of this document to specify how, for each media type, the base URI can be embedded. It is assumed that user agents manipulating such media types will be able to obtain the appropriate syntax from that media type's specification. An example of how the base URI can be embedded in the Hypertext Markup Language (HTML) [RFC1866] is provided in Appendix D.

A mechanism for embedding the base URI within MIME container types (e.g., the message and multipart types) is defined by MHTML [RFC2110]. Protocols that do not use the MIME message header syntax, but which do allow some form of tagged metainformation to be included within messages, may define their own syntax for defining the base URI as part of a message.

5.1.2. Base URI from the Encapsulating Entity

If no base URI is embedded, the base URI of a document is defined by the document's retrieval context. For a document that is enclosed within another entity (such as a message or another document), the retrieval context is that entity; thus, the default base URI of the document is the base URI of the entity in which the document is encapsulated.

5.1.3. Base URI from the Retrieval URI

If no base URI is embedded and the document is not encapsulated within some other entity (e.g., the top level of a composite entity), then, if a URI was used to retrieve the base document, that URI shall be considered the base URI. Note that if the retrieval was the result of a redirected request, the last URI used (i.e., that which resulted in the actual retrieval of the document) is the base URI.

5.1.4. Default Base URI

If none of the conditions described in Sections 5.1.1--5.1.3 apply, then the base URI is defined by the context of the application. Since this definition is necessarily application-dependent, failing

Berners-Lee, et. al.	Standards Track	[Page 19]
□		
RFC 2396	URI Generic Syntax	August 1998

to define the base URI using one of the other methods may result in the same content being interpreted differently by different types of application.

It is the responsibility of the distributor(s) of a document containing relative URI to ensure that the base URI for that document can be established. It must be emphasized that relative URI cannot be used reliably in situations where the document's base URI is not well-defined.

5.2. Resolving Relative References to Absolute Form

This section describes an example algorithm for resolving URI references that might be relative to a given base URI.

The base URI is established according to the rules of Section 5.1 and parsed into the four main components as described in Section 3. Note that only the scheme component is required to be present in the base URI; the other components may be empty or undefined. A component is undefined if its preceding separator does not appear in the URI reference; the path component is never undefined, though it may be empty. The base URI's query component is not used by the resolution algorithm and may be discarded.

For each URI reference, the following steps are performed in order:

- 1) The URI reference is parsed into the potential four components and fragment identifier, as described in Section 4.3.
- 2) If the path component is empty and the scheme, authority, and query components are undefined, then it is a reference to the current document and we are done. Otherwise, the reference URI's query and fragment components are defined as found (or not found) within the URI reference and not inherited from the base URI.
- 3) If the scheme component is defined, indicating that the reference starts with a scheme name, then the reference is interpreted as an absolute URI and we are done. Otherwise, the reference URI's scheme is inherited from the base URI's scheme component.

Due to a loophole in prior specifications [RFC1630], some parsers allow the scheme name to be present in a relative URI if it is the same as the base URI scheme. Unfortunately, this can conflict with the correct parsing of non-hierarchical URI. For backwards compatibility, an implementation may work around such references by removing the scheme if it matches that of the base URI and the scheme is known to always use the <hier_part> syntax. The parser

can then continue with the steps below for the remainder of the reference components. Validating parsers should mark such a malformed relative reference as an error.

- 4) If the authority component is defined, then the reference is a network-path and we skip to step 7. Otherwise, the reference URI's authority is inherited from the base URI's authority component, which will also be undefined if the URI scheme does not use an authority component.
- 5) If the path component begins with a slash character ("/"), then the reference is an absolute-path and we skip to step 7.
- 6) If this step is reached, then we are resolving a relative-path reference. The relative path needs to be merged with the base URI's path. Although there are many ways to do this, we will

describe a simple method using a separate string buffer.

- a) All but the last segment of the base URI's path component is copied to the buffer. In other words, any characters after the last (right-most) slash character, if any, are excluded.
- b) The reference's path component is appended to the buffer string.
- c) All occurrences of "./", where "." is a complete path segment, are removed from the buffer string.
- d) If the buffer string ends with "." as a complete path segment, that "." is removed.
- e) All occurrences of "<segment>/../", where <segment> is a complete path segment not equal to "..", are removed from the buffer string. Removal of these path segments is performed iteratively, removing the leftmost matching pattern on each iteration, until no matching pattern remains.
- f) If the buffer string ends with "<segment>/..", where <segment> is a complete path segment not equal to "..", that "<segment>/.." is removed.
- g) If the resulting buffer string still begins with one or more complete path segments of "..", then the reference is considered to be in error. Implementations may handle this error by retaining these components in the resolved path (i.e., treating them as part of the final URI), by removing them from the resolved path (i.e., discarding relative levels above the root), or by avoiding traversal of the reference.

Berners-Lee, et. al.

Standards Track

[Page 21]

□

RFC 2396

URI Generic Syntax

August 1998

- h) The remaining buffer string is the reference URI's new path component.
- 7) The resulting URI components, including any inherited from the base URI, are recombined to give the absolute form of the URI reference. Using pseudocode, this would be

```

result = ""

if scheme is defined then
    append scheme to result
    append ":" to result

if authority is defined then
    append "://" to result
    append authority to result

append path to result

if query is defined then

```

```

    append "?" to result
    append query to result

    if fragment is defined then
        append "#" to result
        append fragment to result

    return result

```

Note that we must be careful to preserve the distinction between a component that is undefined, meaning that its separator was not present in the reference, and a component that is empty, meaning that the separator was present and was immediately followed by the next component separator or the end of the reference.

The above algorithm is intended to provide an example by which the output of implementations can be tested -- implementation of the algorithm itself is not required. For example, some systems may find it more efficient to implement step 6 as a pair of segment stacks being merged, rather than as a series of string pattern replacements.

Note: Some WWW client applications will fail to separate the reference's query component from its path component before merging the base and reference paths in step 6 above. This may result in a loss of information if the query component contains the strings `"../"` or `"/./"`.

Resolution examples are provided in Appendix C.

Berners-Lee, et. al.	Standards Track	[Page 22]
□		
RFC 2396	URI Generic Syntax	August 1998

6. URI Normalization and Equivalence

In many cases, different URI strings may actually identify the identical resource. For example, the host names used in URL are actually case insensitive, and the URL `<http://www.XEROX.com>` is equivalent to `<http://www.xerox.com>`. In general, the rules for equivalence and definition of a normal form, if any, are scheme dependent. When a scheme uses elements of the common syntax, it will also use the common syntax equivalence rules, namely that the scheme and hostname are case insensitive and a URL with an explicit `":port"`, where the port is the default for the scheme, is equivalent to one where the port is elided.

7. Security Considerations

A URI does not in itself pose a security threat. Users should beware that there is no general guarantee that a URL, which at one time located a given resource, will continue to do so. Nor is there any guarantee that a URL will not locate a different resource at some later point in time, due to the lack of any constraint on how a given authority apportions its namespace. Such a guarantee can only be obtained from the person(s) controlling that namespace and the resource in question. A specific URI scheme may include additional semantics, such as name persistence, if those semantics are required

of all naming authorities for that scheme.

It is sometimes possible to construct a URL such that an attempt to perform a seemingly harmless, idempotent operation, such as the retrieval of an entity associated with the resource, will in fact cause a possibly damaging remote operation to occur. The unsafe URL is typically constructed by specifying a port number other than that reserved for the network protocol in question. The client unwittingly contacts a site that is in fact running a different protocol. The content of the URL contains instructions that, when interpreted according to this other protocol, cause an unexpected operation. An example has been the use of a gopher URL to cause an unintended or impersonating message to be sent via a SMTP server.

Caution should be used when using any URL that specifies a port number other than the default for the protocol, especially when it is a number within the reserved space.

Care should be taken when a URL contains escaped delimiters for a given protocol (for example, CR and LF characters for telnet protocols) that these are not unescaped before transmission. This might violate the protocol, but avoids the potential for such

Berners-Lee, et. al.

Standards Track

[Page 23]

□

RFC 2396

URI Generic Syntax

August 1998

characters to be used to simulate an extra operation or parameter in that protocol, which might lead to an unexpected and possibly harmful remote operation to be performed.

It is clearly unwise to use a URL that contains a password which is intended to be secret. In particular, the use of a password within the 'userinfo' component of a URL is strongly disrecommended except in those rare cases where the 'password' parameter is intended to be public.

8. Acknowledgements

This document was derived from RFC 1738 [RFC1738] and RFC 1808 [RFC1808]; the acknowledgements in those specifications still apply. In addition, contributions by Gisle Aas, Martin Beet, Martin Duerst, Jim Gettys, Martijn Koster, Dave Kristol, Daniel LaLiberte, Foteos Macrides, James Marshall, Ryan Moats, Keith Moore, and Lauren Wood are gratefully acknowledged.

9. References

[RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", BCP 18, RFC 2277, January 1998.

[RFC1630] Berners-Lee, T., "Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web", RFC 1630, June 1994.

- [RFC1738] Berners-Lee, T., Masinter, L., and M. McCahill, Editors,
"Uniform Resource Locators (URL)", RFC 1738, December 1994.
- [RFC1866] Berners-Lee T., and D. Connolly, "HyperText Markup Language
Specification -- 2.0", RFC 1866, November 1995.
- [RFC1123] Braden, R., Editor, "Requirements for Internet Hosts --
Application and Support", STD 3, RFC 1123, October 1989.
- [RFC822] Crocker, D., "Standard for the Format of ARPA Internet Text
Messages", STD 11, RFC 822, August 1982.
- [RFC1808] Fielding, R., "Relative Uniform Resource Locators", RFC
1808, June 1995.
- [RFC2046] Freed, N., and N. Borenstein, "Multipurpose Internet Mail
Extensions (MIME) Part Two: Media Types", RFC 2046,
November 1996.

Berners-Lee, et. al.	Standards Track	[Page 24]
□		
RFC 2396	URI Generic Syntax	August 1998

- [RFC1736] Kunze, J., "Functional Recommendations for Internet
Resource Locators", RFC 1736, February 1995.
- [RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.
- [RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities",
STD 13, RFC 1034, November 1987.
- [RFC2110] Palme, J., and A. Hopmann, "MIME E-mail Encapsulation of
Aggregate Documents, such as HTML (MHTML)", RFC 2110, March
1997.
- [RFC1737] Sollins, K., and L. Masinter, "Functional Requirements for
Uniform Resource Names", RFC 1737, December 1994.
- [ASCII] US-ASCII. "Coded Character Set -- 7-bit American Standard
Code for Information Interchange", ANSI X3.4-1986.
- [UTF-8] Yergeau, F., "UTF-8, a transformation format of ISO 10646",
RFC 2279, January 1998.

Berners-Lee, et. al.

Standards Track

[Page 25]

□

RFC 2396

URI Generic Syntax

August 1998

10. Authors' Addresses

Tim Berners-Lee
World Wide Web Consortium
MIT Laboratory for Computer Science, NE43-356
545 Technology Square
Cambridge, MA 02139

Fax: +1(617)258-8682
EMail: timbl@w3.org

Roy T. Fielding
Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425

Fax: +1(949)824-1715
EMail: fielding@ics.uci.edu

Larry Masinter
Xerox PARC
3333 Coyote Hill Road
Palo Alto, CA 94034

Fax: +1(415)812-4333
EMail: masinter@parc.xerox.com

Berners-Lee, et. al.

Standards Track

[Page 26]

□

RFC 2396

URI Generic Syntax

August 1998

A. Collected BNF for URI

```

URI-reference = [ absoluteURI | relativeURI ] [ "#" fragment ]
absoluteURI   = scheme ":" ( hier_part | opaque_part )
relativeURI   = ( net_path | abs_path | rel_path ) [ "?" query ]

hier_part     = ( net_path | abs_path ) [ "?" query ]
opaque_part   = uric_no_slash *uric

uric_no_slash = unreserved | escaped | ";" | "?" | ":" | "@" |
                "&" | "=" | "+" | "$" | ","

net_path      = "//" authority [ abs_path ]
abs_path      = "/" path_segments
rel_path      = rel_segment [ abs_path ]

rel_segment   = 1*( unreserved | escaped |
                    ";" | "@" | "&" | "=" | "+" | "$" | "," )

scheme        = alpha *( alpha | digit | "+" | "-" | "." )

authority     = server | reg_name

reg_name      = 1*( unreserved | escaped | "$" | "," |
                    ";" | ":" | "@" | "&" | "=" | "+" )

server        = [ [ userinfo "@" ] hostport ]
userinfo      = *( unreserved | escaped |
                    ";" | ":" | "&" | "=" | "+" | "$" | "," )

hostport      = host [ ":" port ]
host          = hostname | IPv4address
hostname      = *( domainlabel "." ) toplabel [ "." ]
domainlabel   = alphanum | alphanum *( alphanum | "-" ) alphanum
toplabel      = alpha | alpha *( alphanum | "-" ) alphanum
IPv4address   = 1*digit "." 1*digit "." 1*digit "." 1*digit
port          = *digit

path          = [ abs_path | opaque_part ]
path_segments = segment *( "/" segment )

```

```

segment      = *pchar *( ";" param )
param        = *pchar
pchar        = unreserved | escaped |
               ":" | "@" | "&" | "=" | "+" | "$" | ","

query        = *uric

fragment     = *uric

```

Berners-Lee, et. al.

Standards Track

[Page 27]

□

RFC 2396

URI Generic Syntax

August 1998

```

uric          = reserved | unreserved | escaped
reserved      = ";" | "/" | "?" | ":" | "@" | "&" | "=" | "+" |
               "$" | ","
unreserved    = alphanum | mark
mark          = "-" | "_" | "." | "!" | "~" | "*" | "'" |
               "(" | ")"

escaped       = "%" hex hex
hex           = digit | "A" | "B" | "C" | "D" | "E" | "F" |
               "a" | "b" | "c" | "d" | "e" | "f"

alphanum      = alpha | digit
alpha         = lowalpha | upalpha

lowalpha      = "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" |
               "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" |
               "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z"
upalpha       = "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" |
               "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
               "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"
digit         = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" |
               "8" | "9"

```

B. Parsing a URI Reference with a Regular Expression

As described in Section 4.3, the generic URI syntax is not sufficient to disambiguate the components of some forms of URI. Since the "greedy algorithm" described in that section is identical to the disambiguation method used by POSIX regular expressions, it is natural and commonplace to use a regular expression for parsing the potential four components and fragment identifier of a URI reference.

The following line is the regular expression for breaking-down a URI reference into its components.

```

^(((12[^:/?#]+)3:4(5/(6[^/?#]*)7)(8[^?#]*)9(9\?([8#]*)9)?)(8(9[^#]*)9)?)(8(9[^#]*)9)?)(8(9[^#]*)9)?

```

The numbers in the second line above are only to assist readability; they indicate the reference points for each subexpression (i.e., each paired parenthesis). We refer to the value matched for subexpression <n> as \$<n>. For example, matching the above expression to

```
http://www.ics.uci.edu/pub/ietf/uri/#Related
```

results in the following subexpression matches:

```

$1 = http:
$2 = http
$3 = //www.ics.uci.edu
$4 = www.ics.uci.edu
$5 = /pub/ietf/uri/
$6 = <undefined>
$7 = <undefined>
$8 = #Related
$9 = Related

```

where <undefined> indicates that the component is not present, as is the case for the query component in the above example. Therefore, we can determine the value of the four components and fragment as

```

scheme      = $2
authority   = $4
path        = $5
query       = $7
fragment    = $9

```

and, going in the opposite direction, we can recreate a URI reference from its components using the algorithm in step 7 of Section 5.2.

C. Examples of Resolving Relative URI References

Within an object with a well-defined base URI of

`http://a/b/c/d;p?q`

the relative URI would be resolved as follows:

C.1. Normal Examples

<code>g:h</code>	<code>= g:h</code>
<code>g</code>	<code>= http://a/b/c/g</code>
<code>./g</code>	<code>= http://a/b/c/g</code>
<code>g/</code>	<code>= http://a/b/c/g/</code>
<code>/g</code>	<code>= http://a/g</code>
<code>//g</code>	<code>= http://g</code>
<code>?y</code>	<code>= http://a/b/c/?y</code>
<code>g?y</code>	<code>= http://a/b/c/g?y</code>
<code>#s</code>	<code>= (current document)#s</code>
<code>g#s</code>	<code>= http://a/b/c/g#s</code>
<code>g?y#s</code>	<code>= http://a/b/c/g?y#s</code>
<code>;x</code>	<code>= http://a/b/c/x</code>
<code>g;x</code>	<code>= http://a/b/c/g;x</code>
<code>g;x?y#s</code>	<code>= http://a/b/c/g;x?y#s</code>
<code>.</code>	<code>= http://a/b/c/</code>
<code>./</code>	<code>= http://a/b/c/</code>
<code>..</code>	<code>= http://a/b/</code>
<code>../</code>	<code>= http://a/b/</code>
<code>../g</code>	<code>= http://a/b/g</code>
<code>../..</code>	<code>= http://a/</code>
<code>../..//</code>	<code>= http://a/</code>
<code>../..//g</code>	<code>= http://a/g</code>

C.2. Abnormal Examples

Although the following abnormal examples are unlikely to occur in normal practice, all URI parsers should be capable of resolving them consistently. Each example uses the same base as above.

An empty reference refers to the start of the current document.

<code><></code>	<code>= (current document)</code>
-----------------------	-----------------------------------

Parsers must be careful in handling the case where there are more relative path `".."` segments than there are hierarchical levels in the base URI's path. Note that the `".."` syntax cannot be used to change the authority component of a URI.

□

```

../../g      = http://a../g
../../../../g = http://a../../../../g

```

In practice, some implementations strip leading relative symbolic elements (".", "..") after applying a relative URI calculation, based on the theory that compensating for obvious author errors is better than allowing the request to fail. Thus, the above two references will be interpreted as "http://a/g" by some implementations.

Similarly, parsers must avoid treating "." and ".." as special when they are not complete components of a relative path.

```

./g          = http://a./g
../g         = http://a../g
g.           = http://a/b/c/g.
.g           = http://a/b/c/.g
g..          = http://a/b/c/g..
..g          = http://a/b/c/..g

```

Less likely are cases where the relative URI uses unnecessary or nonsensical forms of the "." and ".." complete path segments.

```

../g         = http://a/b/g
./g/.        = http://a/b/c/g/
g./h         = http://a/b/c/g/h
g../h        = http://a/b/c/h
g;x=1/./y    = http://a/b/c/g;x=1/y
g;x=1/../y   = http://a/b/c/y

```

All client applications remove the query component from the base URI before resolving relative URI. However, some applications fail to separate the reference's query and/or fragment components from a relative path before merging it with the base path. This error is rarely noticed, since typical usage of a fragment never includes the hierarchy ("/") character, and the query component is not normally used within relative references.

```

g?y/./x      = http://a/b/c/g?y/./x
g?y/../x     = http://a/b/c/g?y/../x
g#s/./x      = http://a/b/c/g#s/./x
g#s/../x     = http://a/b/c/g#s/../x

```

□

Some parsers allow the scheme name to be present in a relative URI if it is the same as the base URI scheme. This is considered to be a loophole in prior specifications of partial URI [RFC1630]. Its use should be avoided.

```
http:g      = http:g      ; for validating parsers
              | http://a/b/c/g ; for backwards compatibility
```

D. Embedding the Base URI in HTML documents

It is useful to consider an example of how the base URI of a document can be embedded within the document's content. In this appendix, we

describe how documents written in the Hypertext Markup Language (HTML) [RFC1866] can include an embedded base URI. This appendix does not form a part of the URI specification and should not be considered as anything more than a descriptive example.

HTML defines a special element "BASE" which, when present in the "HEAD" portion of a document, signals that the parser should use the BASE element's "HREF" attribute as the base URI for resolving any relative URI. The "HREF" attribute must be an absolute URI. Note that, in HTML, element and attribute names are case-insensitive. For example:

```
<!doctype html public "-//IETF//DTD HTML//EN">
<HTML><HEAD>
<TITLE>An example HTML document</TITLE>
<BASE href="http://www.ics.uci.edu/Test/a/b/c">
</HEAD><BODY>
... <A href="../x">a hypertext anchor</A> ...
</BODY></HTML>
```

A parser reading the example document should interpret the given relative URI "../x" as representing the absolute URI

```
<http://www.ics.uci.edu/Test/a/x>
```

regardless of the context in which the example document was obtained.

E. Recommendations for Delimiting URI in Context

URI are often transmitted through formats that do not provide a clear context for their interpretation. For example, there are many occasions when URI are included in plain text; examples include text sent in electronic mail, USENET news messages, and, most importantly, printed on paper. In such cases, it is important to be able to delimit the URI from the rest of the text, and in particular from

punctuation marks that might be mistaken for part of the URI.

In practice, URI are delimited in a variety of ways, but usually within double-quotes "http://test.com/", angle brackets <http://test.com/>, or just using whitespace

http://test.com/

These wrappers do not form part of the URI.

In the case where a fragment identifier is associated with a URI reference, the fragment would be placed within the brackets as well (separated from the URI with a "#" character).

In some cases, extra whitespace (spaces, linebreaks, tabs, etc.) may need to be added to break long URI across lines. The whitespace should be ignored when extracting the URI.

No whitespace should be introduced after a hyphen ("-") character. Because some typesetters and printers may (erroneously) introduce a hyphen at the end of line when breaking a line, the interpreter of a URI containing a line break immediately after a hyphen should ignore all unescaped whitespace around the line break, and should be aware that the hyphen may or may not actually be part of the URI.

Using <> angle brackets around each URI is especially recommended as a delimiting style for URI that contain whitespace.

The prefix "URL:" (with or without a trailing space) was recommended as a way to used to help distinguish a URL from other bracketed designators, although this is not common in practice.

For robustness, software that accepts user-typed URI should attempt to recognize and strip both delimiters and embedded whitespace.

For example, the text:

Berners-Lee, et. al.	Standards Track	[Page 34]
□		
RFC 2396	URI Generic Syntax	August 1998

Yes, Jim, I found it under "http://www.w3.org/Addressing/", but you can probably pick it up from <ftp://ds.internic.net/rfc/>. Note the warning in <http://www.ics.uci.edu/pub/ietf/uri/historical.html#WARNING>.

contains the URI references

http://www.w3.org/Addressing/
 ftp://ds.internic.net/rfc/
 http://www.ics.uci.edu/pub/ietf/uri/historical.html#WARNING

Berners-Lee, et. al.	Standards Track	[Page 35]
□		
RFC 2396	URI Generic Syntax	August 1998

F. Abbreviated URLs

The URL syntax was designed for unambiguous reference to network resources and extensibility via the URL scheme. However, as URL identification and usage have become commonplace, traditional media (television, radio, newspapers, billboards, etc.) have increasingly used abbreviated URL references. That is, a reference consisting of only the authority and path portions of the identified resource, such as

`www.w3.org/Addressing/`

or simply the DNS hostname on its own. Such references are primarily intended for human interpretation rather than machine, with the assumption that context-based heuristics are sufficient to complete the URL (e.g., most hostnames beginning with "www" are likely to have

a URL prefix of "http://"). Although there is no standard set of heuristics for disambiguating abbreviated URL references, many client implementations allow them to be entered by the user and heuristically resolved. It should be noted that such heuristics may change over time, particularly when new URL schemes are introduced.

Since an abbreviated URL has the same syntax as a relative URL path, abbreviated URL references cannot be used in contexts where relative URLs are expected. This limits the use of abbreviated URLs to places where there is no defined base URL, such as dialog boxes and off-line advertisements.

Berners-Lee, et. al.

Standards Track

[Page 36]

□

RFC 2396

URI Generic Syntax

August 1998

G. Summary of Non-editorial Changes

G.1. Additions

Section 4 (URI References) was added to stem the confusion regarding "what is a URI" and how to describe fragment identifiers given that they are not part of the URI, but are part of the URI syntax and parsing concerns. In addition, it provides a reference definition for use by other IETF specifications (HTML, HTTP, etc.) that have previously attempted to redefine the URI syntax in order to account for the presence of fragment identifiers in URI references.

Section 2.4 was rewritten to clarify a number of misinterpretations and to leave room for fully internationalized URI.

Appendix F on abbreviated URLs was added to describe the shortened references often seen on television and magazine advertisements and explain why they are not used in other contexts.

G.2. Modifications from both RFC 1738 and RFC 1808

Changed to URI syntax instead of just URL.

Confusion regarding the terms "character encoding", the URI "character set", and the escaping of characters with %<hex><hex> equivalents has (hopefully) been reduced. Many of the BNF rule names regarding the character sets have been changed to more accurately describe their purpose and to encompass all "characters" rather than just US-ASCII octets. Unless otherwise noted here, these modifications do not affect the URI syntax.

Both RFC 1738 and RFC 1808 refer to the "reserved" set of characters as if URI-interpreting software were limited to a single set of characters with a reserved purpose (i.e., as meaning something other than the data to which the characters correspond), and that this set was fixed by the URI scheme. However, this has not been true in practice; any character that is interpreted differently when it is escaped is, in effect, reserved. Furthermore, the interpreting engine on a HTTP server is often dependent on the resource, not just the URI scheme. The description of reserved characters has been changed accordingly.

The plus "+", dollar "\$", and comma "," characters have been added to those in the "reserved" set, since they are treated as reserved within the query component.

Berners-Lee, et. al.

Standards Track

[Page 37]

□

RFC 2396

URI Generic Syntax

August 1998

The tilde "~" character was added to those in the "unreserved" set, since it is extensively used on the Internet in spite of the difficulty to transcribe it with some keyboards.

The syntax for URI scheme has been changed to require that all schemes begin with an alpha character.

The "user:password" form in the previous BNF was changed to a "userinfo" token, and the possibility that it might be "user:password" made scheme specific. In particular, the use of passwords in the clear is not even suggested by the syntax.

The question-mark "?" character was removed from the set of allowed characters for the userinfo in the authority component, since testing showed that many applications treat it as reserved for separating the query component from the rest of the URI.

The semicolon ";" character was added to those stated as being reserved within the authority component, since several new schemes are using it as a separator within userinfo to indicate the type of user authentication.

RFC 1738 specified that the path was separated from the authority portion of a URI by a slash. RFC 1808 followed suit, but with a

fudge of carrying around the separator as a "prefix" in order to describe the parsing algorithm. RFC 1630 never had this problem, since it considered the slash to be part of the path. In writing this specification, it was found to be impossible to accurately describe and retain the difference between the two URI

<foo:/bar> and <foo:bar>

without either considering the slash to be part of the path (as corresponds to actual practice) or creating a separate component just to hold that slash. We chose the former.

G.3. Modifications from RFC 1738

The definition of specific URL schemes and their scheme-specific syntax and semantics has been moved to separate documents.

The URL host was defined as a fully-qualified domain name. However, many URLs are used without fully-qualified domain names (in contexts for which the full qualification is not necessary), without any host (as in some file URLs), or with a host of "localhost".

The URL port is now *digit instead of 1*digit, since systems are expected to handle the case where the ":" separator between host and port is supplied without a port.

Berners-Lee, et. al.	Standards Track	[Page 38]
□		
RFC 2396	URI Generic Syntax	August 1998

The recommendations for delimiting URI in context (Appendix E) have been adjusted to reflect current practice.

G.4. Modifications from RFC 1808

RFC 1808 (Section 4) defined an empty URL reference (a reference containing nothing aside from the fragment identifier) as being a reference to the base URL. Unfortunately, that definition could be interpreted, upon selection of such a reference, as a new retrieval action on that resource. Since the normal intent of such references is for the user agent to change its view of the current document to the beginning of the specified fragment within that document, not to make an additional request of the resource, a description of how to correctly interpret an empty reference has been added in Section 4.

The description of the mythical Base header field has been replaced with a reference to the Content-Location header field defined by MHTML [RFC2110].

RFC 1808 described various schemes as either having or not having the properties of the generic URI syntax. However, the only requirement is that the particular document containing the relative references have a base URI that abides by the generic URI syntax, regardless of the URI scheme, so the associated description has been updated to reflect that.

The BNF term <net_loc> has been replaced with <authority>, since the latter more accurately describes its use and purpose. Likewise, the

authority is no longer restricted to the IP server syntax.

Extensive testing of current client applications demonstrated that the majority of deployed systems do not use the ";" character to indicate trailing parameter information, and that the presence of a semicolon in a path segment does not affect the relative parsing of that segment. Therefore, parameters have been removed as a separate component and may now appear in any path segment. Their influence has been removed from the algorithm for resolving a relative URI reference. The resolution examples in Appendix C have been modified to reflect this change.

Implementations are now allowed to work around misformed relative references that are prefixed by the same scheme as the base URI, but only for schemes known to use the <hier_part> syntax.

Berners-Lee, et. al.	Standards Track	[Page 39]
□		
RFC 2396	URI Generic Syntax	August 1998

H. Full Copyright Statement

Copyright (C) The Internet Society (1998). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Berners-Lee, et. al.

Standards Track

[Page 40]

□

International Conference: 25-27 March 1998, Bristol, UK

IRISS '98: Conference Papers



Proceedings

Title: The Latest Web Developments

Author: Brian Kelly

Summary

This paper outlines some of the latest World Wide Web developments, in particular standards which are emerging from W3C, the World Wide Web Consortium. The paper gives an overview of the architectural components of the Web, summarise their deficiencies and describe how these deficiencies are being addressed.

The paper should be of interest to people involved in developing applications and services on the Web and anyone who has a general interest in future developments of the Web.

Background

The World Wide Web (often referred to as the web) is a distributed hypermedia system which is based on three key architectural components:

1. Data format
2. Addressing
3. Transport

The native file format for resources on the web is the *HyperText Markup Language* (HTML).

The address for resources on the web is given by a *Uniform Resource Locator* (URL).

Resources on the web are transported from a server to the user's client system using the *Hypertext Transport Protocol* (HTTP).

We will look at these three architectural components in more detail.

Data Format

HTML (Hypertext Markup Language) is an application of SGML (Standard Generalised Markup Language). The first release, HTML 1.0, provided the hypertext linking which Web users today will be familiar with. HTML 1.0, in keeping with the spirit of SGML of defining the structural elements in documents, included the basic structural elements still in use today, such as paragraphs (the <P> element) and headings (<H1> to <H6>) as well as a small number of formatting elements, such as italic <I> and bold .

HTML 2.0 introduced a number of innovations which were incorporated in NCSA's Mosaic web browser, including inline images and forms. Yes the initial implementation of the web did not include inline images!

At the first international WWW conference held in CERN, Switzerland in May 1994 David Raggett outlined a roadmap for future developments of HTML. HTML 3.0 (which was initially known as HTML+) would include a range of new features such as tables, richer forms and support for mathematical equations.

HTML 3.0 was submitted to the Internet Engineering Task Force (IETF). Unfortunately it failed to be standardised, due to a failure to reach consensus. This failure was due partly to the size and complexity of the proposal and also due to the lack of interest from the commercial web browser vendors.

In October 1994, the first version of the Netscape browser was released. Although Netscape proved tremendously popular, it also, controversially, announced support for a number of HTML elements which have not featured in discussions of developments to HTML such as the infamous <BLINK> element.

By 1995 Microsoft had become aware of the importance of the web. Initially their browser, Internet Explorer, was based on a licensed version of the original Mosaic browser. By the time Internet Explorer 3.0 was released (which was developed in-house), Microsoft were beginning to compete with Netscape for browser market share. This competition resulted in both companies announcing a variety of new HTML elements, with, for example, Microsoft responding to <BLINK> with their <MARQUEE> element for displaying scrolling text.

The browser wars resulted in confusion within the marketplace. Large companies, who were beginning to invest large sums of money in corporate Intranets, found the lack of interworking across browser and platforms placed a barrier on further growth.

At the same time as large corporations began to express their concerns over the browser wars, developers of web standards began to raise doubts as to the long term effectiveness of what became known as the HTML "tag soup". Pressures from large corporate users on one side and the web standards community on the other helped to force Microsoft and Netscape to work together, within W3C working groups responsible for coordinating the development of new HTML proposals. By January 1997 the HTML 3.2 proposal was accepted as a W3C recommendation [1]. HTML 3.2 was based on current established working practices. During 1997 work began on a new version of HTML, which had the codename *Cougar*. In December 1997 W3C announced [2] that HTML 4.0 (as Cougar became known as) had been accepted as a W3C recommendation.

HTML 4.0 included enhancements in a number of areas, such as more sophisticated forms and tables. HTML 4.0 added features to make web resources more accessible by providing support for people with disabilities and for non-English speaking users. Although HTML 4.0 gave recognition to the widespread deployment of frames, it did not introduce a wide range of new features. It primarily provided hooks for embedding other resources within HTML documents, such as multimedia objects and scripting languages. In addition HTML 4.0 provided support for style sheets.

Style Sheets

As mentioned earlier, HTML was originally intended to define the structure of a document. It has always been recognised that the appearance of a document was important. However it was felt that the

appearance should be held separately from the content of a document.

The initial recommendation for style sheets, Cascading Style Sheets level 1 (CSS1), was announced in December 1996 [3]. However CSS1 was only partly supported in Microsoft's Internet Explorer 3.0 (which was available at the time) and was not supported in Netscape Navigator 3.0. Exeriented gained in the way in which CSS1 was used highlighted a number of backwards-compatibility issues.

In November 1997 A draft release of CSS level 2 was announced [4]. CSS2 provides a great deal of control over the appearance of a document. CSS can be included inline within an HTML element or within the HEAD of a document. However for maintenance purposes, it is better if the CSS is included as an external linked file. For example, all of the conference papers published in the conference proceedings could point to a single style sheet file. Changing the house style for the papers will simply require changing a single file.

An example of use of a simple style sheet is shown in Figure 1.

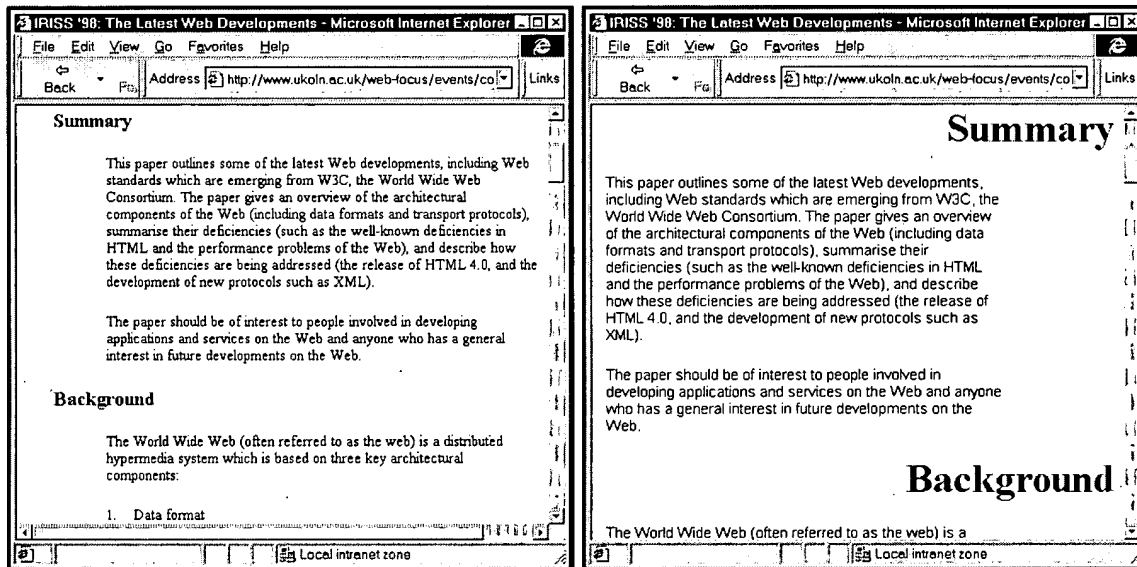


Figure 1a - Simple Style Sheet Example Figure 1b - Simple Style Sheet Example

Figures 1a and 1b show the same document content, with slightly different style sheets. The corresponding style sheets are given below.

```
<STYLE>
<!--
H1, H2 {color: blue; margin-left: 5%}
H3 {margin-left: 10%}
P {margin-left: 15%}
OL {margin-left: 20%}
-->
</STYLE>
```

```
<STYLE>
<!--
H1, H2 {color: red; text-align: right;
        font-size: 24pt}
H3 {margin-left: 5%}
P {margin-left: 5%; margin-right: 20%;
   font-family: arial}
OL {margin-left: 20%}
-->
</STYLE>
```

Note that style sheets do not have to be supplied by an author. It is possible for an end user to define a style sheet to be used when accessing pages.

An example of user-supplied style sheet is shown in Figure 2.

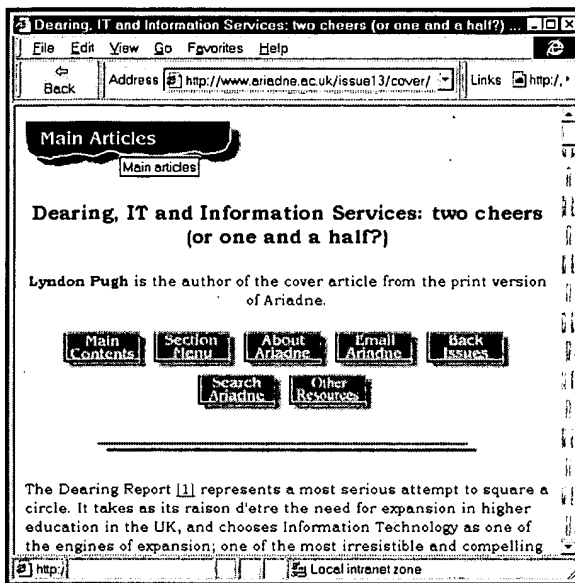


Figure 2a - Original Display

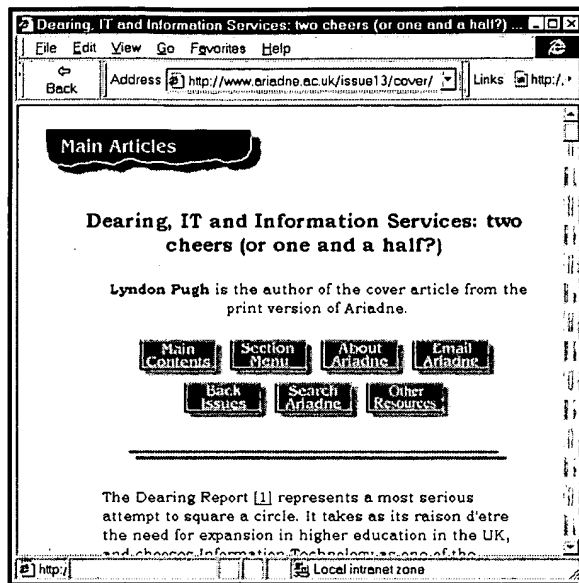


Figure 2b - Using A User-supplied Style Sheet

Figure 2a shows the original document, with the formatting defined by the author. Figure 2b shows the document when viewed using a user-supplied style sheet. In this example the style sheet indents the left-hand margin and specifies a colour for the headings.

Dynamic HTML

HTML 4.0 provides a means of defining the document structure, allowing CSS2 to define how the document appears. *Dynamic HTML* provides a way of enabling the content of HTML and CSS elements to be changed.

Dynamic HTML is based on a *Document Object Model* (DOM) [5] for HTML and CSS elements. The values of the HTML and CSS elements can be changed in response to a user action, such as clicking the mouse or moving the mouse over an object. The changes are initiated using a client-side scripting language, such as JavaScript.

Figure 3 gives an illustration of simple use of Dynamic HTML.

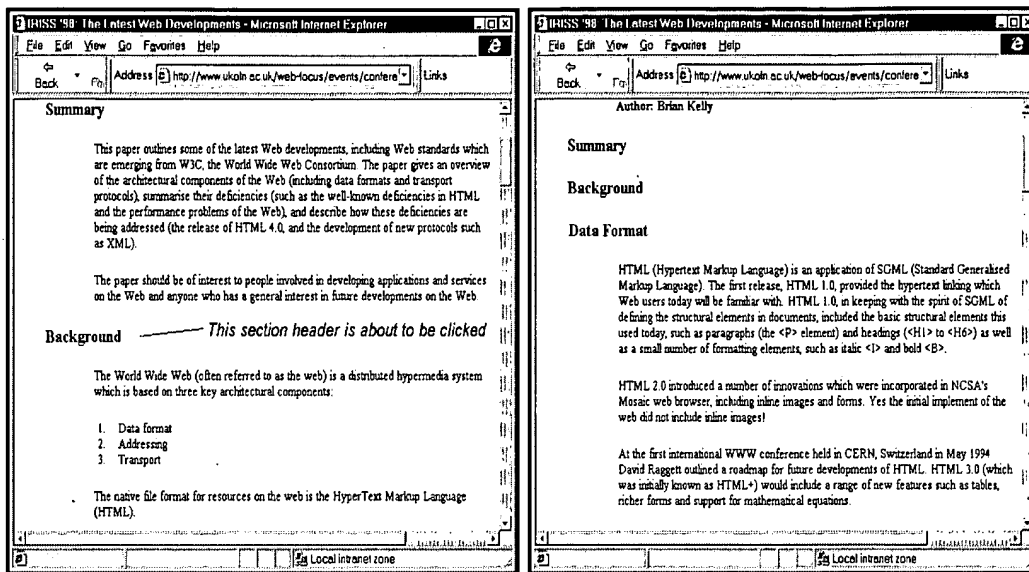


Figure 3a - Original Display

Figure 3b - Display after Headings "Collapsed"

Figure 3a gives the original display. After clicking on the first two headings, the text underneath the headings is collapsed. This is achieved by setting the visibility of the section following the heading to "none" when the heading is clicked.

The code to achieve this effect is simply:

```
<H2 STYLE="cursor:hand" onclick="toggleDisplay(Background);"
onmouseover="this.style.color = 'blue'"
onmouseout="this.style.color = 'black'">Background</H2>
<DIV ID="Background" STYLE="display: visible">
...
</DIV>
```

Two Javascript routines, each of about 10 lines is executed when the heading is clicked which sets the display of the document identified by the name <DIV ID="Background"> on or off. These Javascript routines are not included in this paper. However very simple Dynamic HTML can be seen in the HTML fragment above. The onmouseover="this.style.color = 'blue'" changes the colour of the heading to blue when the mouse is positioned over the heading.

XML

If HTML 4.0 can define the document structure and provide the hooks for including multimedia objects, scripting languages and links to style sheets, CSS2 the appearance of the document and Dynamic HTML can provide a mechanism for dynamically altering HTML and CSS elements, does this mean that work on data formats is complete?

The answer to this is, perhaps not surprisingly, no.

The HTML standardisation process is too slow and time-consuming for new elements to be introduced. For example, if we wanted to introduce a new element called <ABBREVIATION>, even if we could achieve consensus within the HTML developers community, it would still take a long time for the recommendation to be agreed. Even then there would be no certainty that the browser vendors would

provide support for the new element.

In addition, although HTML, in conjunction with CSS2, can be used as an output format it is not sufficiently rich to be used as a data storage format. For example we cannot develop a web application within our institution for storing records such as <STUDENT-NUMBER> or <PART-NUMBER>.

Finally even if communities such as mathematicians or chemists could agree on a set of elements for use within their community, adding them to a new version of HTML would result in a very large, complex language.

The *Extensible Markup Language* (XML) has been developed to address these issues. XML has been designed to be extensible, so that agreement on a set of standard elements does not necessarily have to be achieved. XML can be regarded as a light-weight version of SGML, designed for network use.

Although XML was only announced as a W3C recommendation in February 1998 [6], it is already becoming widely adopted in a number of areas. The Mathematical Markup Language (MathML), which is due to be submitted as a W3C Proposed Recommendation in February 1998, is an XML application, as is the Chemical Markup Language (CML).

In addition to use within the scientific communities, XML is also being used within the web community to develop new architectural components to the web, especially in the area of metadata, as discussed later.

Addressing

The location of a resource on the web is given by its URL. For example the URL of W3C's HTML 4.0 recommendation is <http://www.w3.org/TR/REC-html40>

As can be seen in this example, URLs contain the domain name of the machine together with (in many cases) the location in the underlying directory structure. This can be regarded as equivalent to stating that the hard copy of the specification is located in the MIT library, and it's the sixth book along on the third shelf on the fourth floor.

URLs suffer from their dependency on the location. If an organisation reorganises its website, links to resources are likely to be broken. Similarly if an organisation changes its name, is taken over or sells part of the organisation, a reorganisation of its website to reflect the changes will also result in broken links.

There have been a number of proposals which attempt to provide a location-independent address for a resource including Uniform Resource Names (URNs) [7] and Persistent Uniform Resources (PURLs) [8].

A PURL acts as a URL which points to a resolution services rather than the resource itself. The PURL resolution service associates the PURL with the URL of the resource and uses a HTTP redirect to access the resource.

More recently the Digital Object Identifier (DOI) system has been developed [9]. The DOI system has three components: the identifier, the directory and the database. The system allows identifiers to be assigned at various levels. The directory is a distributed system based on CNRI's Handle system which provides a mapping from DOIs to URLs. DOIs have initially been aimed at the 'traditional' publishing

industry, and there are plans to use the DOI as the basis of copyright management systems.

However none of the proposals for replacing URLs have been widely deployed. This is, in part, due to the need for an organisational infrastructure for registering location-independent resources.

Transport

HTTP, the *HyperText Transfer Protocol*, governs the transfer of resources between a web server and client. Typically clicking on a hypertext link in a web browser will send a HTTP GET request to the server. The web server will then send back a series of headers, together with the resource, if it exists.

It is possible to emulate a web client using telnet, as illustrated below:

```
% telnet www.w3.org 80          Telnet to port 80
GET / HTTP/1.0                 Request the default file
                                Enter blank line
HTTP/1.1 200 OK                Confirmation received
Date: Tue, 24 Feb 1998 09:14:31 GMT Misc headers displayed
Server: Apache/1.2.5
Last-Modified: Fri, 20 Feb 1998 17:55:12 GMT
ETag: "2c3136-23c1-34edc380"
Content-Length: 9153
Accept-Ranges: bytes
Connection: close
Content-Type: text/html; charset=ISO-8859-1
                                HTML document displayed
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
..
```

In the initial implementation of HTTP, HTTP/0.9, the web browser could process the files based on the file suffix. So, for example, a PostScript file with a .ps could be passed to a PostScript viewer for displaying. This, however, was not a scaleable solution. In HTTP/1.0 [10] files are sent as MIME attachments, such as text/html, image/gif, text/postscript, etc.

Although HTTP/1.0 is now being widely used, there are a number of problems with it:

- HTTP/1.0 uses TCP inefficiently. Since most resources are small, and HTTP/1.0 opens and closes a new TCP connection for each operation, there is a large overhead.
- HTTP/1.0 does not have sufficient facilities for compression.
- HTTP/1.0's caching is very primitive.

HTTP/1.1 [11] was developed to address these deficiencies and to fix a number of bugs in HTTP/1.0. The HTTP/1.1 specification provides support for multiple TCP connections and more efficient support for caching.

A W3C Note on "*Network Performance Effects of HTTP/1.1, CSS1, and PNG*" [12] confirms the performance benefits of HTTP/1.1.

Extending HTTP

Although HTTP/1.1 will provide performance benefits, the introduction of new facilities is still hindered by the standardisation process and the dangers of making HTTP more complex by the introduction of facilities which will be used by only small communities. HTTP faces similar development problems as does HTML.

Just as XML provides a extension mechanism for data formats, the Protocol Extension Protocol (PEP) [13] is designed to provide an extension mechanism for HTTP.

PEP examples which are given in the PEP draft specification include determining whether a server understands and supports the Distributed Authoring and Versioning (DAV) protocol extension and use of a micropayments scheme.

An example of the potential use of PEP is a micropayments system for accessing resources. The dialogue is illustrated below.

```
GET /Index HTTP/1.1
Host: some.host

420 Policy Not Fulfilled
PEP-Info: {{id "http://www.w3.org/PEP/MiniPayment"}
           {params {Price 0.02USD}} {strength must}}

PEP-GET /Index HTTP/1.1
Host: some.host
PEP: {{map "http://www.w3.org/PEP/MiniPayment" 12-}
      {strength must}} 12-Price: 0.02USD

HTTP/1.1 200 OK
```

In the example given above the client requests a resource. The server responds with an HTTP response code stating that the policy has not been fulfilled, and then uses the PEP extension mechanism to state a price which must be paid in order to access the resource, together with the address describing the minipayment protocol. A web client which does not understand the PEP request will treat the response as a file not found and display an appropriate error message. Otherwise the client can communicate with the server using the extension policy.

Content Negotiation

We have seen how PEP can be used to provide an extension mechanism for HTTP. Transparent Content Negotiation (TCN) [14] provides an extensible negotiation mechanism, layered on top of HTTP, for automatically selecting the "best" version when the resource is accessed. TCN enables new data formats and HTML elements to be smoothly deployed.

HTTP/NG

Although HTTP/1.1, together with PEP and TCN, are addressing a number of the deficiencies in the underlying transport protocol, we are still faced with a number of problem areas, including the complexity of HTTP, the poor scalability of HTTP when faced with today's network traffic load and the difficulty of introducing applications on the web, other than simple document retrieval applications.

HTTP/NG [15] will be a new architecture for the HTTP protocol based on a simple, extensible distributed object-oriented model. Work on HTTP/NG started recently. As yet there is little information

publicly available.

Other Areas

Metadata can be regarded as the missing architectural component of the web. Although HTML has allowed the basic elements of a document structure to be defined, it has, in general, not allowed information about the document to be defined in a structured, machine-parsable way.

The <META> HTML element was an initial attempt to provide a mechanism for storing document *metadata* in a standard way. The <META> element became popular for storing keywords to assist search engines, such as Alta Vista, in finding resources. Search engines would give a high priority to resources containing metadata as shown below:

```
<META NAME="description" VALUE="This is the HTTP specification">  
<META NAME="keywords" VALUE="HTTP, web, transport protocol">
```

Dublin Core [16] is the name given to an initiative to agree a common, core set of metadata attributes to help with resource discovery. The Dublin Core now consists of 15 elements, such as Title, Creator, Date, etc. Initially attempts were made to embed Dublin Core metadata using the <META> element. However this was not sufficiently flexible to cater for more complex use of Dublin Core metadata, such as hierarchical structures, such as the creators name, postal address, email address, etc.

The development of a more general solution to the provision of metadata is being coordinated by the W3C. The Resource Description Framework (RDF) [17] is designed to provide an infrastructure to support metadata in a range of areas including resource discovery, sitemaps, rating schemes, and collections of resources.

What's All This Mean For Me?

This document has described a number of developments to web protocols. But what are the implications for web administrators, support staff, software developers, information providers and end users?

The strict HTML philosophy has been to encourage authors to define document structure. With the release of CSS2 and support for CSS2 by the current versions of both the popular browsers, it is now possible for authors to provide a pleasing design for their resources, using technologies which will minimise future maintenance.

Unfortunately Internet Explorer and Netscape have different implementation of style sheets, and so authors will have to make use of these new features with care. It is, possible, however, to layer new technologies, such as CSS and Dynamic HTML on to existing resources, provided the resources conform to standards.

Developers of computer aided learning software, who in the past have made use of proprietary, platform-specific authoring tools will appreciate the development of Dynamic HTML and the Document Object Model. This should enable rich interactive teaching systems to be developed based on open standards.

There are a number of implications for support staff responsible for providing and supporting web software, including web browsers and servers. For performance reasons servers should be upgraded to

support HTTP/1.1. This should not prove too difficult as there are likely to be only a small numbers of web servers within an institution. Upgrading of web browsers to support new developments such as HTML 4.0 and CSS2, may be more difficult. However developments such as Transparent Content Negotiation may make it possible to deploy new features without disenfranchising large communities.

Developments to addressing have not progressed as rapidly as those to data formats. It appears unlikely that we will see in the near future the widespread use of location-independent identifiers. Authors will therefore have to continue to think long and hard about their directory naming conventions, to ensure that next year's reorganisation of a web site does not result in lots of broken links.

References

1. W3C Issues Recommendation for HTML 3.2, W3C Press Release, <URL: <http://www.w3.org/Press/HTML32-REC-PR.html>>
2. The World Wide Web Consortium Issues HTML 4.0 as a W3C Recommendation, W3C Press Release, <URL: <http://www.w3.org/Press/HTML4-REC>>
3. The World Wide Web Consortium Issues Recommendation for CSS1, W3C Press Release, <URL: <http://www.w3.org/Press/CSS1-REC-PR.html>>
4. The World Wide Web Consortium Publishes Public Draft for CSS2, W3C Press Release, <URL: <http://www.w3.org/Press/CSS2>>
5. Document Object Model , W3C, <URL: <http://www.w3.org/DOM/>>
6. The World Wide Web Consortium Issues XML 1.0 as a W3C Recommendation, W3C Press Release, <URL: <http://www.w3.org/Press/1998/XML10-REC>>
7. Naming and Addressing: URIs, W3C, <URL: <http://www.w3.org/Addressing/>>
8. PURLs, OCLC, <URL: <http://purl.oclc.org/>>
9. DOI System, <URL: <http://www.doi.org/>>
10. HTTP/1.0, W3C, <URL: <http://www.ics.uci.edu/pub/ietf/http/rfc1945>>
11. HTTP/1.1, W3C, <URL: <http://www.w3.org/Protocols/rfc2068/rfc2068>>
12. Network Performance Effects of HTTP/1.1, CSS1, and PNG, W3C, <URL: <http://www.w3.org/Protocols/HTTP/Performance/Pipeline.html>>
13. PEP, W3C, <URL: <http://www.w3.org/Protocols/PEP/>>
14. Transparent Content Negotiation in HTTP, Koen Holtman, <URL: <http://gewis.win.tue.nl/~koen/conneg/>>
15. HTTP/NG, W3C, <URL: <http://www.w3.org/Protocols/HTTP-NG/>>
16. Dublin Core, OCLC, <URL: http://purl.org/metadata/dublin_core>
17. RDF, W3, <URL: <http://www.w3.org/Metadata/RDF>>

The Author

Brian Kelly is UK Web Focus - a national web coordination post for the UK Higher Education community. Brian is based at UKOLN (UK Office for Library and Information Networking), University of Bath. His responsibilities include monitoring web developments, keeping the UK HE community informed of web developments, coordinating various web activities within the community and representing JISC on W3C (the World Wide Web Consortium).

Brian has been involved with the Web since early 1993. He helped set up the Web service at Leeds University in January 1993 - the first institutional web service in the UK HE community. He was active in promoting the web throughout the community, giving numerous presentations. He attended the first WWW conference in Switzerland in May 1994, and gave a paper on *Providing Information On The World Wide Web* at the JENC 6 / INET 94 conference in Prague in June 1994.

From October 1995 to October 1996 Brian worked as the Senior Netskills Trainer for the Netskills project, based at Newcastle University. He moved to UKOLN in November 1996.



[Themes](#) [Home](#) [Proceedings](#)

IRISS '98: Last updated: 22 April 1998

*IRISS'98 Conference Office: Institute for Learning and Research Technology, University of Bristol, 8
Woodland Road, Bristol BS8 1TN*

Telephone: +44 (0)117 928 8472/4/8 : Fax: +44 (0)117 928 8473 : email: iriss-info@bris.ac.uk

Patent Storm

[Home](#) [Browse by Inventor](#) [Browse by Date](#) [Links](#) [Contact Us](#)

Type your search term here



United States Patent 7120862

Today In History

March 7, 1876
Alexander Graham
patent for the teleph

Method and apparatus for persistent access to Web resources using variable time-stamps

US Patent Issued on October 10, 2006

Inventor(s)

[Ping-Wen Ong](#)

Assignee

[Lucent Technologies Inc.](#)

Application

No. 09342408 filed on 1999-06-28

Current US Class

[715/511](#), [707/203](#), [707/3](#), [715/501.1](#),
[715/513](#)

Examiners

[Primary: Stephen Hong](#)

[Assistant: Thu V. Huynh](#)

Attorney, Agent or Firm

US Patent References

[5559991](#)
[5819273](#)
[5832224](#)
[5832478](#)
[5907837](#)
[5943678](#)
[5946699](#)
[5978847](#)
[5991773](#)
[5991802](#)
[6006227](#)
[6125371](#)
[6138141](#)

[ABSTRACT](#) [CLAIMS](#) [DESCRIPTION](#) [FULL TEXT](#)

[Ads by Goooooogle](#)

[Advertise on this site](#)

ProfessionalPatentSearch

We use the latest technologies to give you an information edge

[www.PlanetPatent.com](#)

Patent Valuations Expert

Precise trademark & patent valuations. Submit your RFI.

[EricksonPartnersLLC.com](#)

Abstract

A method and apparatus are disclosed for providing persistent storage of Web resources. Uniform Resource Locators (URLs) that identify Web resources are augmented to include a time stamp. A web browser and a web server are disclosed that accommodate a time stamp parameter and allow a user to refer to any Web address with a precise target date. The disclosed Web browser can optionally include a mechanism to facilitate the specification of the desired date and time, or the user can manually append the time stamp to the URL indicated in the "Location" window of the browser. The persistent Web servers (i) receive URLs containing a time stamp, relative or variable time-stamp, (ii) extract the time stamp, (iii) retrieve the appropriate Web page(s) corresponding to the time-stamp, and (iv) return the

6163778 appropriate page(s) or links to the client. Wildcard characters and date ranges can be used in the variable time-stamp to implement a variable time stamp when a user is not sure of the date for a specific web resource or wishes to specify more than one precise date and time. The persistent Web servers include a persistent archive for storing all of the versions of Web resources that will be persistently available to Web users.

Other References

â□□How to Compose a Searchâ□□ (herein after Compose Search), copyright 1997, pp. 1-2.

â□□Search the Kolb-Proust Archive Documentsâ□□ (herein after Kolb-Proust Archive, <http://gateway.library.uiuc.edu/kolbp/Search1.html>, copyright 1997, pp. 1-16.

â□□Welcome to The Libertarian Web!â□□ print out (hereinafter Libertarian) is surfed using <http://www.archive.org>, published Nov. 9, 1996, pp. 1-11.

â□□Building a digital library for the futureâ□□ print out (hereinafter Archive97) is surfed using <http://www.archive.org>, published Jan. 26, 1997, pp. 1-21.

Plan 9 FAQ, Bell Labs and Toronto University, downloaded from <http://www.ecf.toronto.edu/plan9/plan9faq.html> (1995).

Plan 9, Bell Labs, Lucent Technologies, Inc., downloaded from <http://cm.bell-labs.com/plan9> (1998).

Keith Shafer et al., â□□Introduction to Persistent Uniform Resource Locators,â□□ downloaded from <http://purl.oclc.org/OCLC/PURL/INET96> (1996).

K. Sollins, â□□Architectural Principles of Uniform Resource Name Resolution,â□□ RFC2276, downloaded from <http://www.cis.ohio-state.edu/htbin/rfc/rfc2276.html> (Jan. 1998).

Uniform Resource Names, Los Alamos National Laboratory, downloaded from <http://www.acl.lanl.gov/URN> (Aug. 1996).

â□□Uniform Resource Names: A progress Report,â□□ D-Lib Magazine (Feb. 1996).

K. Sollins and L. Masinter, â□□Functional Requirements for Uniform Resource Names,â□□ RFC1737, downloaded from <http://www.cis.ohio-state.edu/htbin/rfc/rfc1737.html> (Dec. 1994).

Archive97, â□□Building a Digital Library for the Futureâ□□ downloaded from <http://www.archive.org>, 1-21 (Jan. 26. 1997).

Brewster Kahle, â□□Archiving the Internetâ□□ Scientific American, 1-8 (Nov. 4, 1996).

Simonson et al., "Version Augmented URIs for Reference Permanence via an Apache Module Design," Computer Networks and ISDN Systems, vol. 30, No. 1-7, 337-345 (Apr. 1998).

Stuart Weibel et al., "PURLs: Persistent Uniform Resource Locators," downloaded from <http://purl.oclc.org/OCLC/OURL/SUMMARY>(not date).

[Home](#) | [Browse by Inventor](#) | [Browse by Date](#) | [Resources](#) | [Contact Us](#)

© 2004-6 PatentStorm LLC. All rights reserved.